US006546014B1

(12) **United States Patent**
Kramer et al.

(10) **Patent No.: US 6,546,014 B1**
(45) **Date of Patent: Apr. 8, 2003**

(54) **METHOD AND SYSTEM FOR DYNAMIC BANDWIDTH ALLOCATION IN AN OPTICAL ACCESS NETWORK**

(75) Inventors: **Glen Kramer**, Davis, CA (US); **Gerry Pesavento**, Davis, CA (US)

(73) Assignee: **Alloptic, Inc.**, Livermore, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 85 days.

(21) Appl. No.: **09/759,539**

(22) Filed: **Jan. 12, 2001**

(51) **Int. Cl.**[7] **H04L 12/56**
(52) **U.S. Cl.** **370/395.41**; 370/449
(58) **Field of Search** 370/395.4, 395.41, 370/395.42, 395.43, 422, 426, 438, 439, 447, 448, 461, 462, 449, 450, 451, 452, 453, 455, 458, 230

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,623,668	A		4/1997	Nieuwenhuizen
5,648,958	A	*	7/1997	Counterman 370/437
5,794,117	A		8/1998	Benard
5,917,822	A	*	6/1999	Lyles et al. 370/395.4
5,978,374	A		11/1999	Ghaibeh et al.
6,088,360	A	*	7/2000	Amaral et al. 348/385.1
2001/0030785	A1	*	10/2001	Pangrac et al. 359/125
2001/0038620	A1	*	11/2001	Stanwood et al. 370/336

OTHER PUBLICATIONS

Alan Quayle, Broadband Passive Optical Network Media Access Control Protocols, SPIE Proceedings vol. 2919, 1996, pp. 268–278.

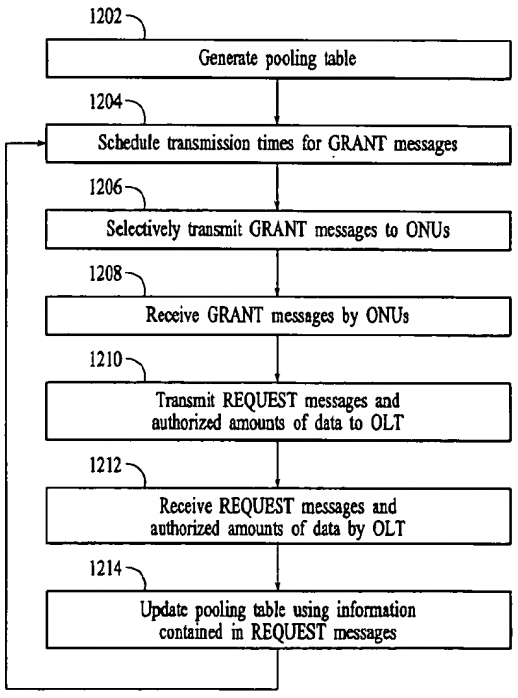
* cited by examiner

Primary Examiner—Kwang Bin Yao
(74) *Attorney, Agent, or Firm*—Wilson & Ham; Thomas Ham

(57) **ABSTRACT**

An optical access network and method for transmitting optical data in the network utilizes an interleaved polling scheme to efficiently use the available bandwidth of the network. The use of the interleaved polling scheme allows a central terminal of the network to dynamically allocate upstream bandwidth from remote terminals of the network to the central terminal in response to the amount of data that is waiting at the remote terminals to be transmitted to the OLT. In one embodiment, the optical access network is based on Passive Optical Network (PON) technology. In another embodiment, the optical access network utilizes Ethernet protocol to encapsulate data in Ethernet frames for transmission. Thus, in these embodiments, the optical access network includes all of the advantages associated with the PON technology and/or the Ethernet protocol. In addition, since the allocation of upstream bandwidth is on an as needed basis, loss of bandwidth due to unfilled time slots is substantially eliminated.

43 Claims, 12 Drawing Sheets



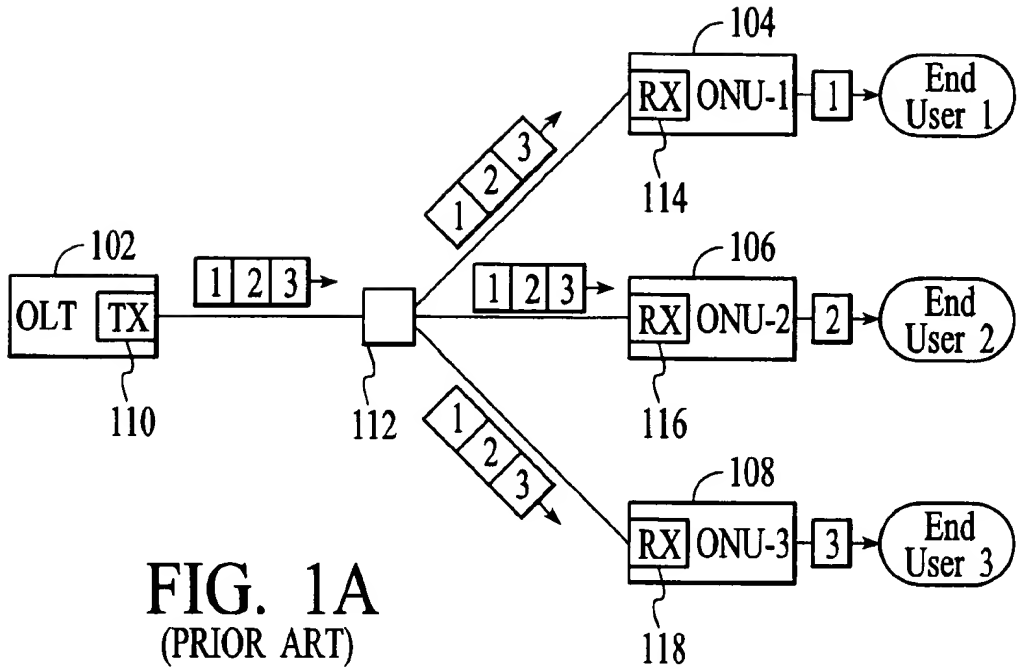


FIG. 1A
(PRIOR ART)

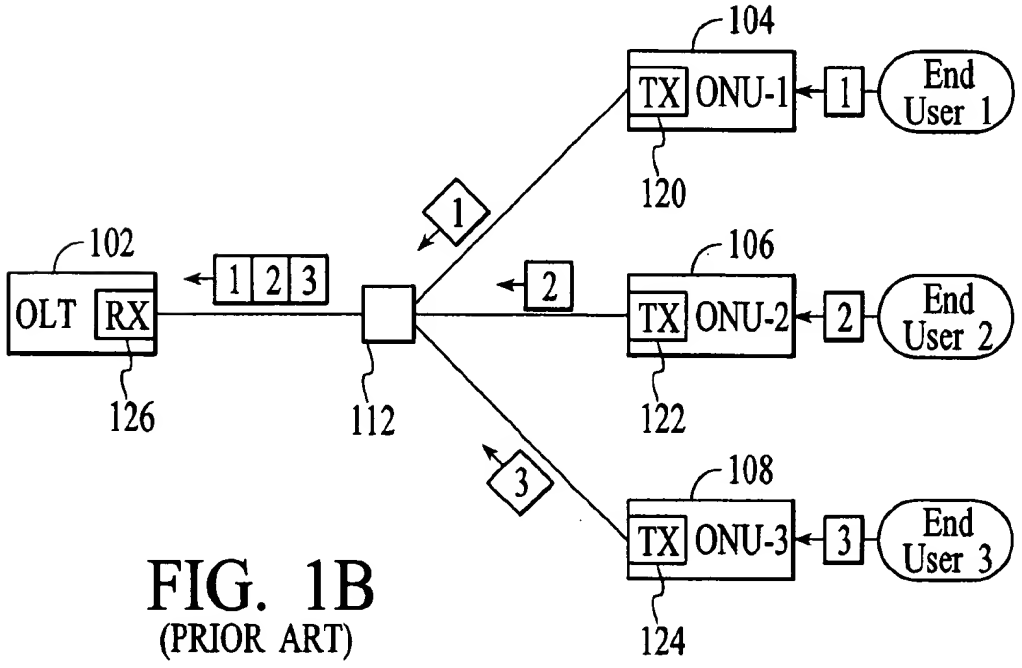


FIG. 1B
(PRIOR ART)

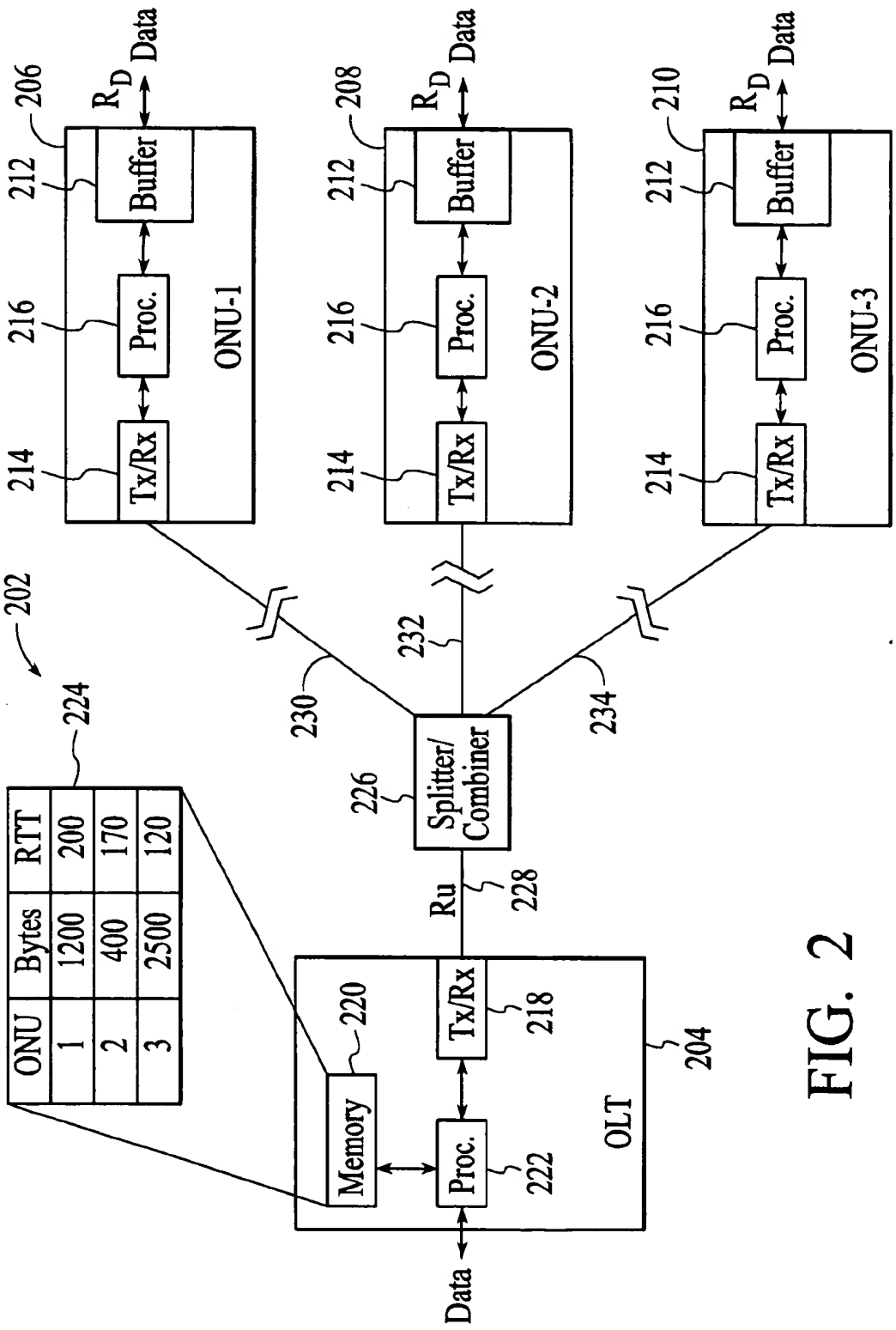

206
208
210
R_D
Data
212
Buffer
216
Proc.
214
Tx/Rx
ONU-1
ONU-2
ONU-3
202
224
230
232
234
226
Splitter/
Combiner
Ru
228
ONU
Bytes
RTT
1
1200
200
2
400
170
3
2500
120
220
Memory
218
222
OLT
204
Data

FIG. 2

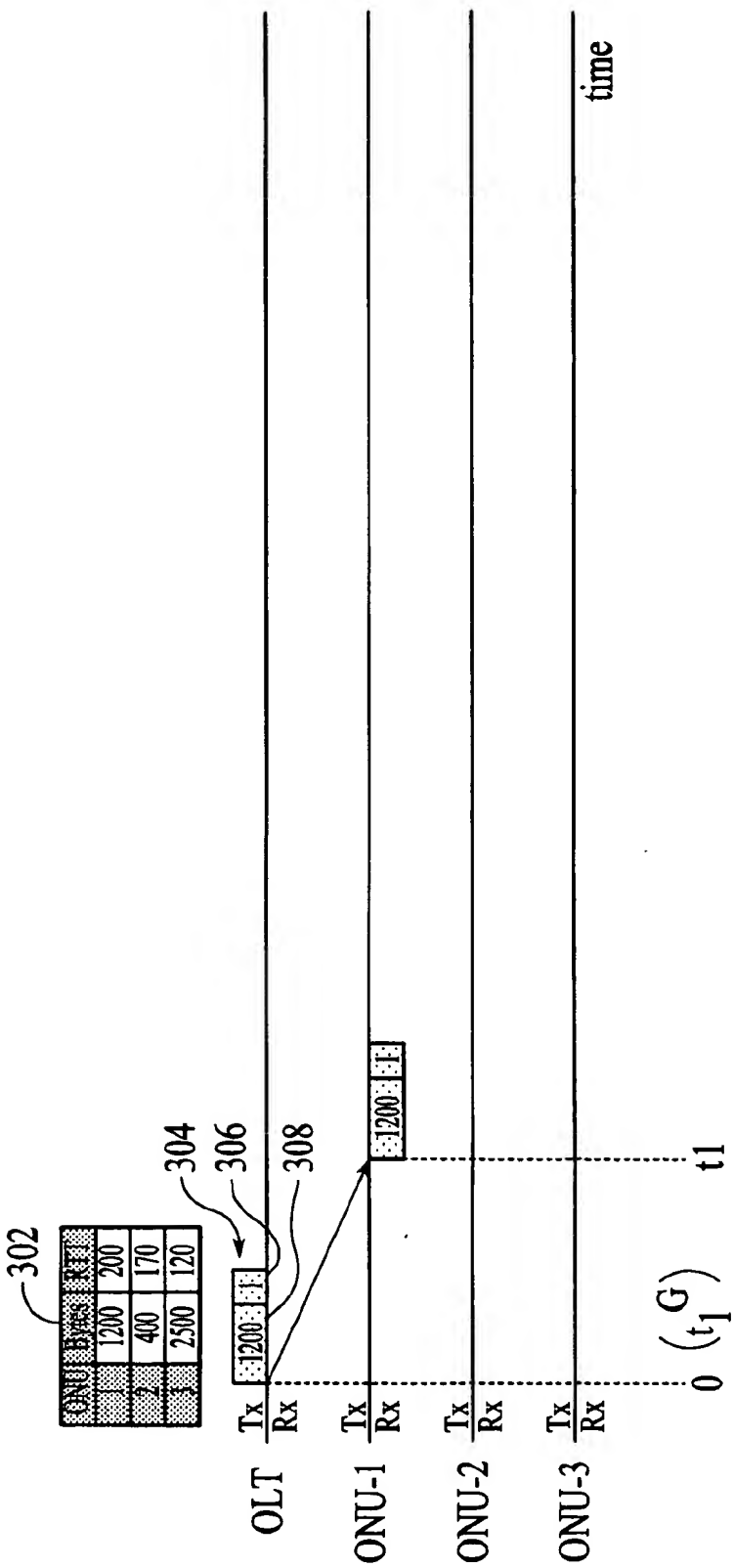

302
ONU
Bytes
RT
1
1200
200
2
400
170
3
2500
120
304
306
308
1200
1200
Tx
Rx
OLT
ONU-1
ONU-2
ONU-3
time
0
t1
(t_1^G)

FIG. 3

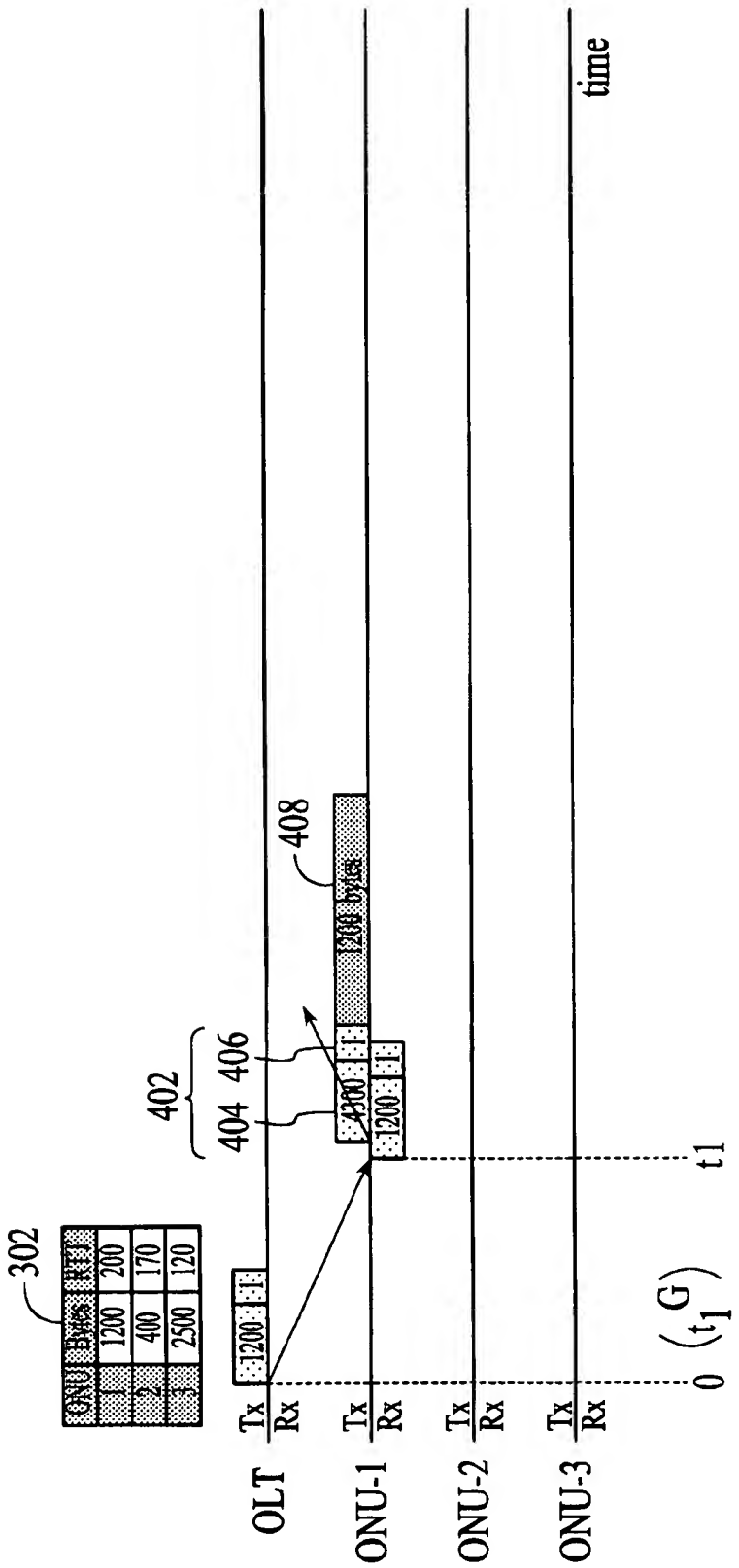

302
ONU
Bytes
RTT
1
1200
200
2
400
170
3
2500
120
402
404
406
408
1200 bytes
2300
1200
OLT
ONU-1
ONU-2
ONU-3
Tx
Rx
time
0
t1
(t1 G)

FIG. 4

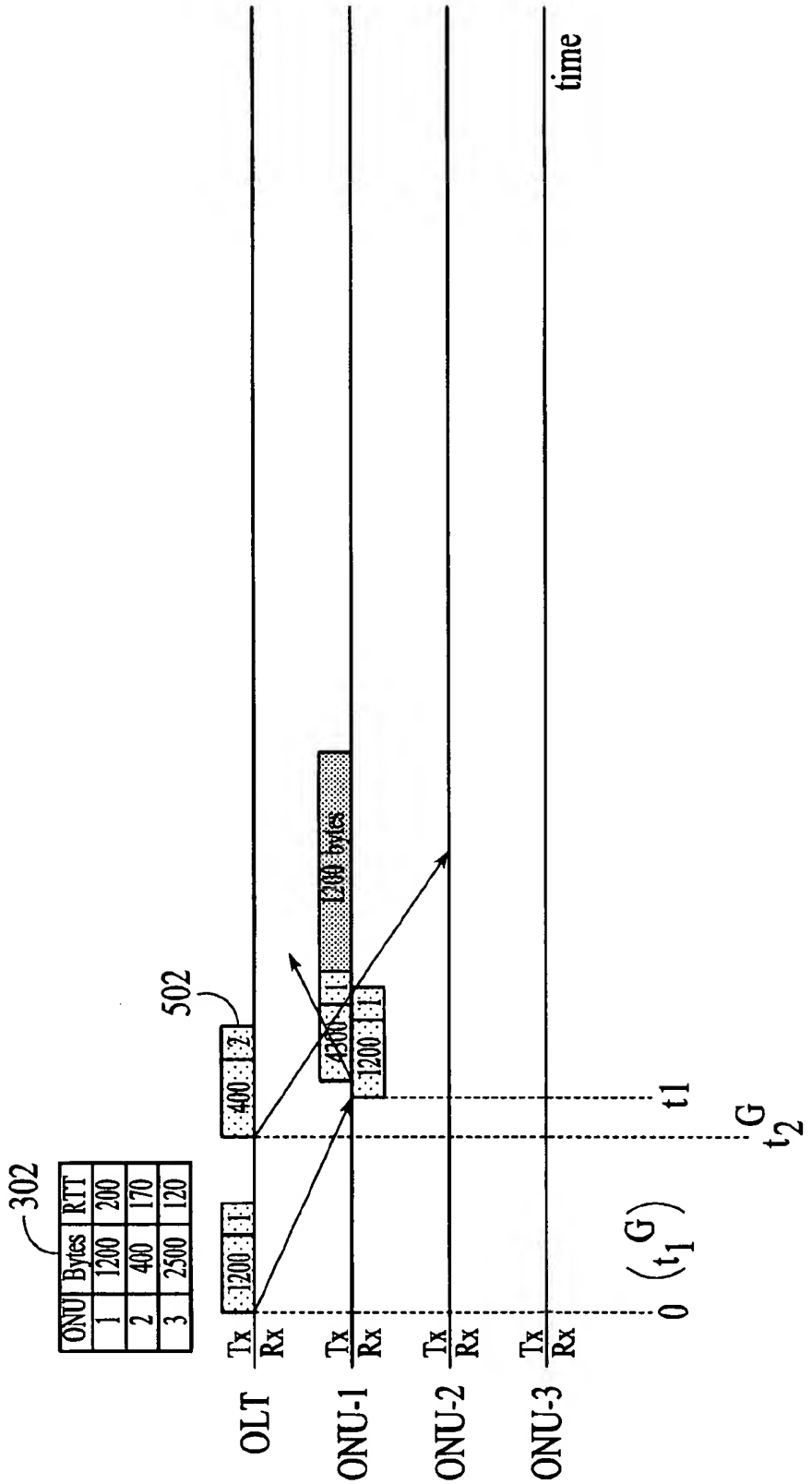

time
OLT
ONU-1
ONU-2
ONU-3
Tx
Rx
0 (t_1^G)
t1
t_2^G
1200
400
1200 bytes
1200
502
302
ONU Bytes RTT
1 1200 200
2 400 170
3 2500 120

FIG. 5

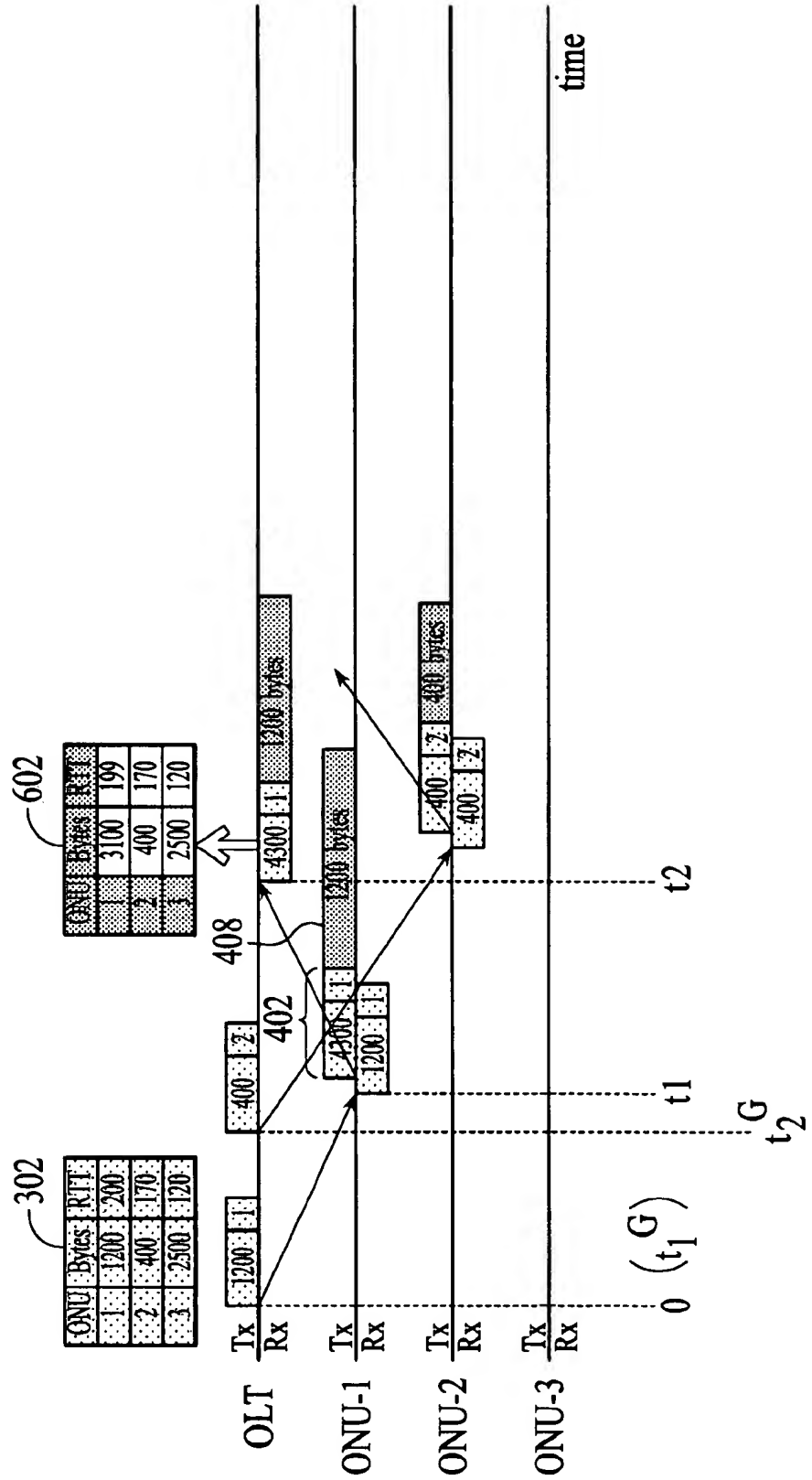

time
OLT
ONU-1
ONU-2
ONU-3
Tx
Rx
302
602
408
402
ONU Bytes RTT
1 1200 200
2 400 170
3 2500 120
1 3100 199
2 400 170
3 2500 120
1200 bytes
400 bytes
t1
t2
G
0 (t1 G)

FIG. 6

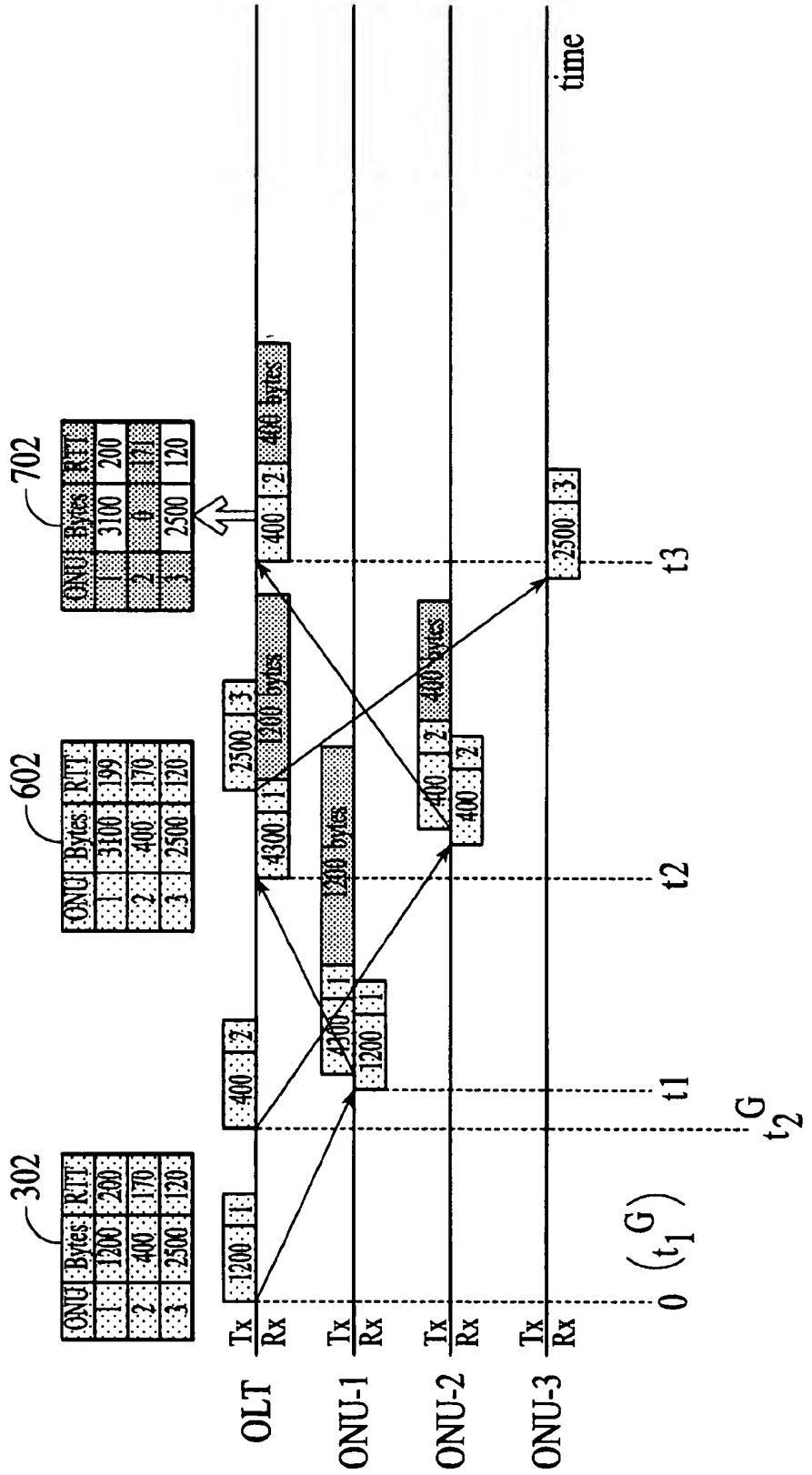

302
ONU Bytes RTT
1 1200 200
2 400 170
3 2500 120
602
ONU Bytes RTT
1 3100 199
2 400 170
3 2500 120
702
ONU Bytes RTT
1 3100 200
2 0 171
3 2500 120
OLT
ONU-1
ONU-2
ONU-3
Tx
Rx
time
0
t1
t2
t3
G
t2
(t1 G)
1200 bytes
400 bytes
2500 bytes
4300 bytes

FIG. 7

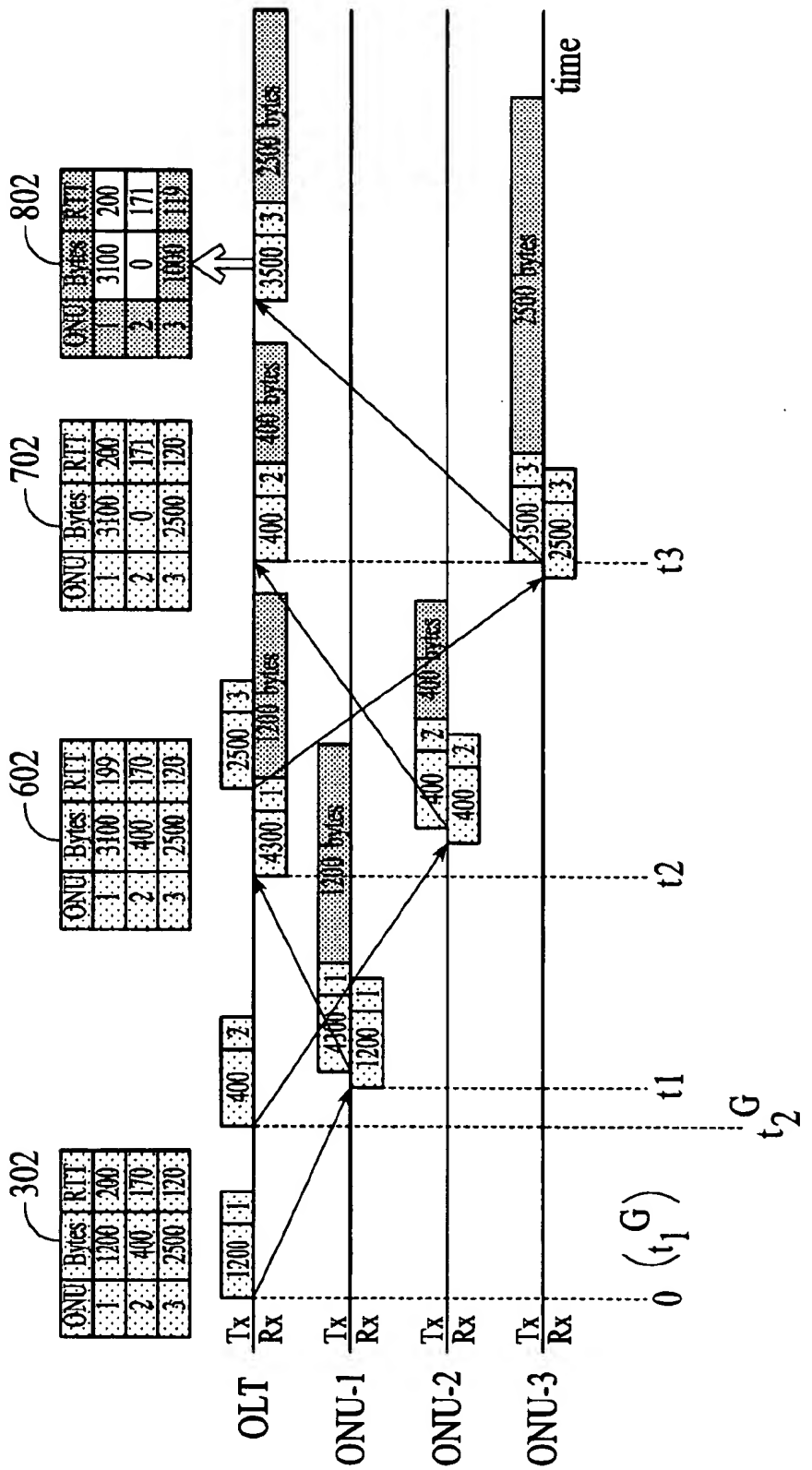

302
ONU Bytes RTT
1 1200 200
2 400 170
3 2500 120
602
ONU Bytes RTT
1 3100 199
2 400 170
3 2500 120
702
ONU Bytes RTT
1 3100 200
2 0 171
3 2500 120
802
ONU Bytes RTT
1 3100 200
2 0 171
3 1000 119
OLT
ONU-1
ONU-2
ONU-3
Tx
Rx
time
0
(t_1^G)
t1
t2
t3
t_2^G
1200 bytes
400 bytes
4300 bytes
1900 bytes
2500 bytes
3500 bytes

FIG. 8

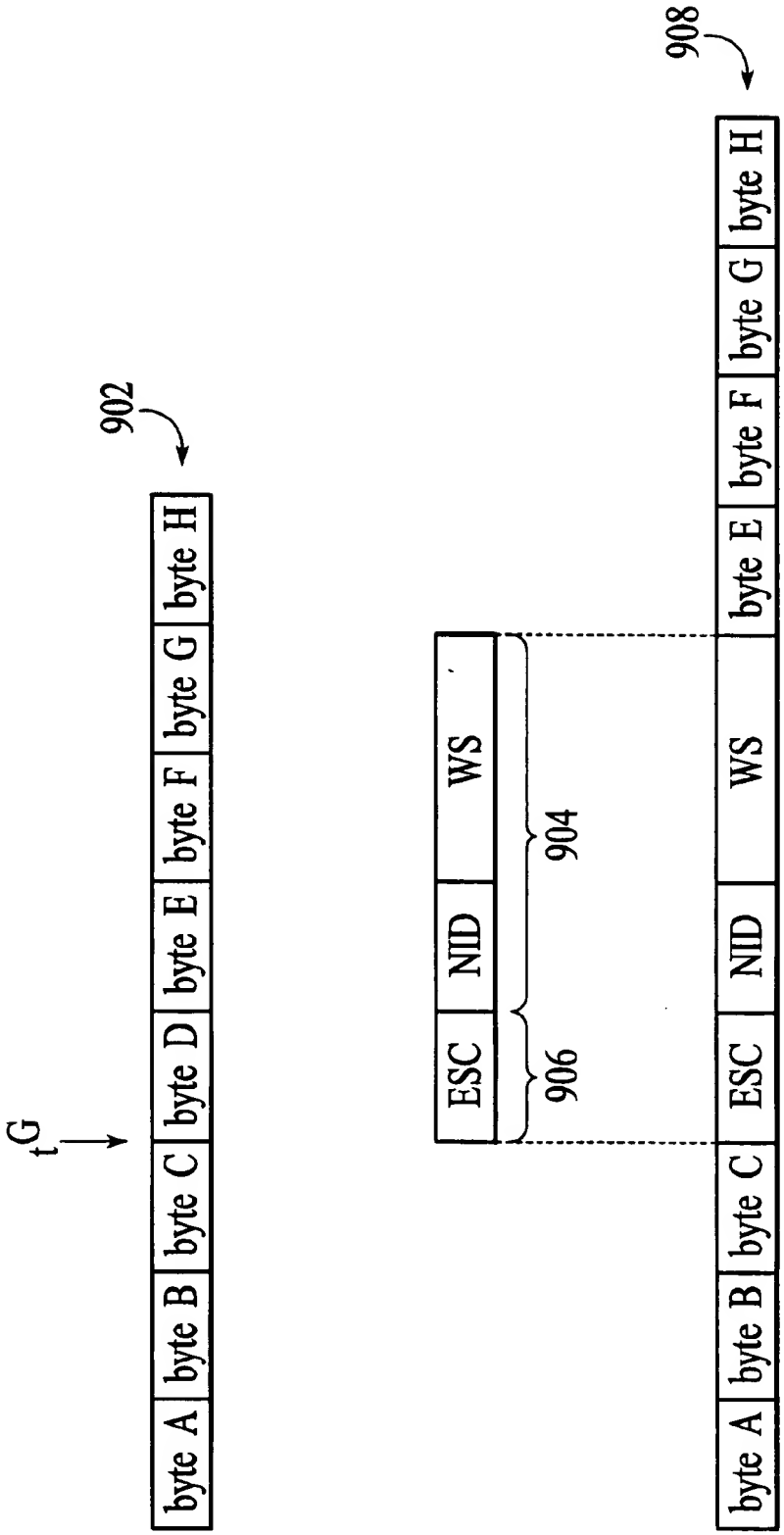

902
908
byte A
byte B
byte C
byte D
byte E
byte F
byte G
byte H
tG
ESC
NID
WS
904
906

FIG. 9

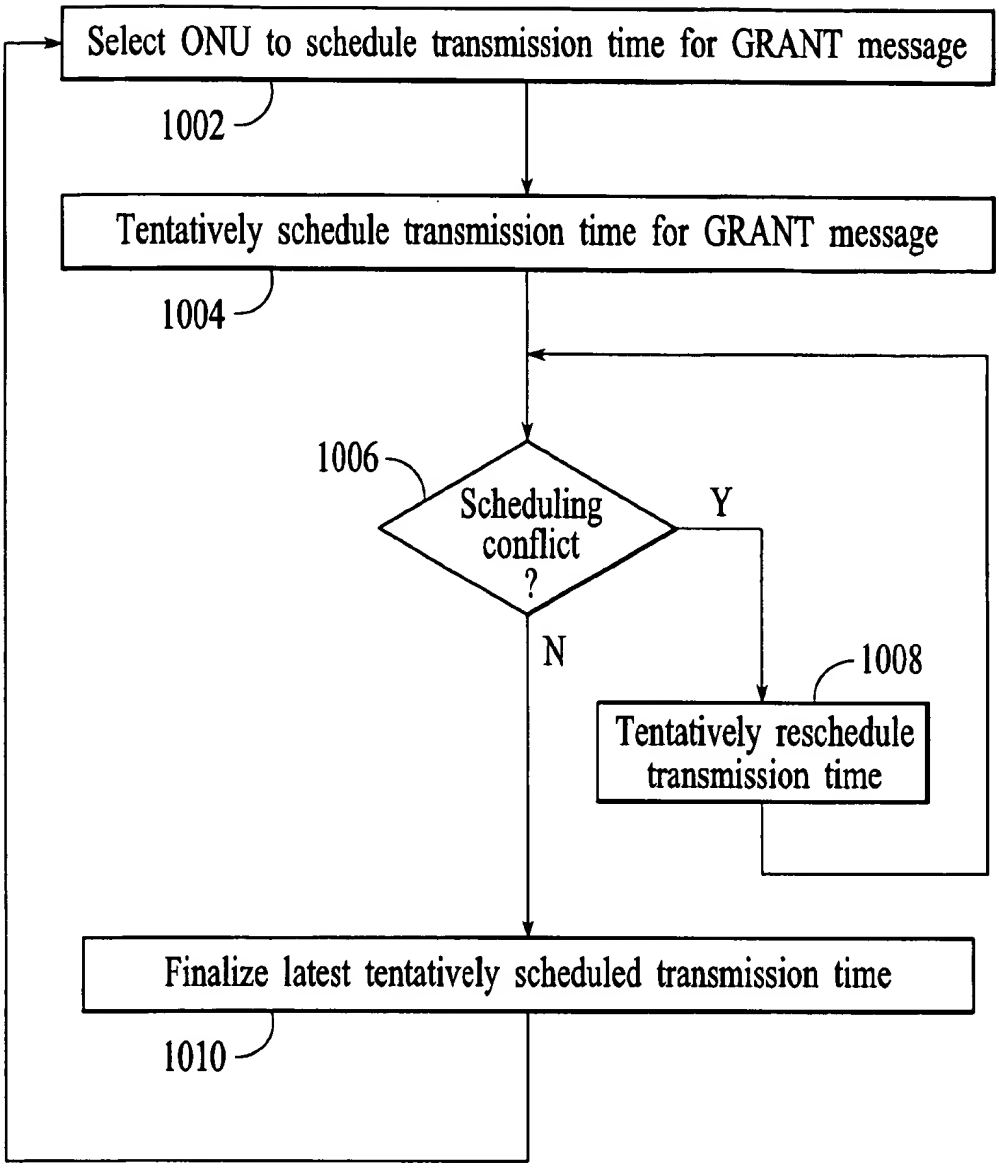

Select ONU to schedule transmission time for GRANT message
1002
Tentatively schedule transmission time for GRANT message
1004
1006
Scheduling conflict ?
Y
N
1008
Tentatively reschedule transmission time
Finalize latest tentatively scheduled transmission time
1010

FIG. 10

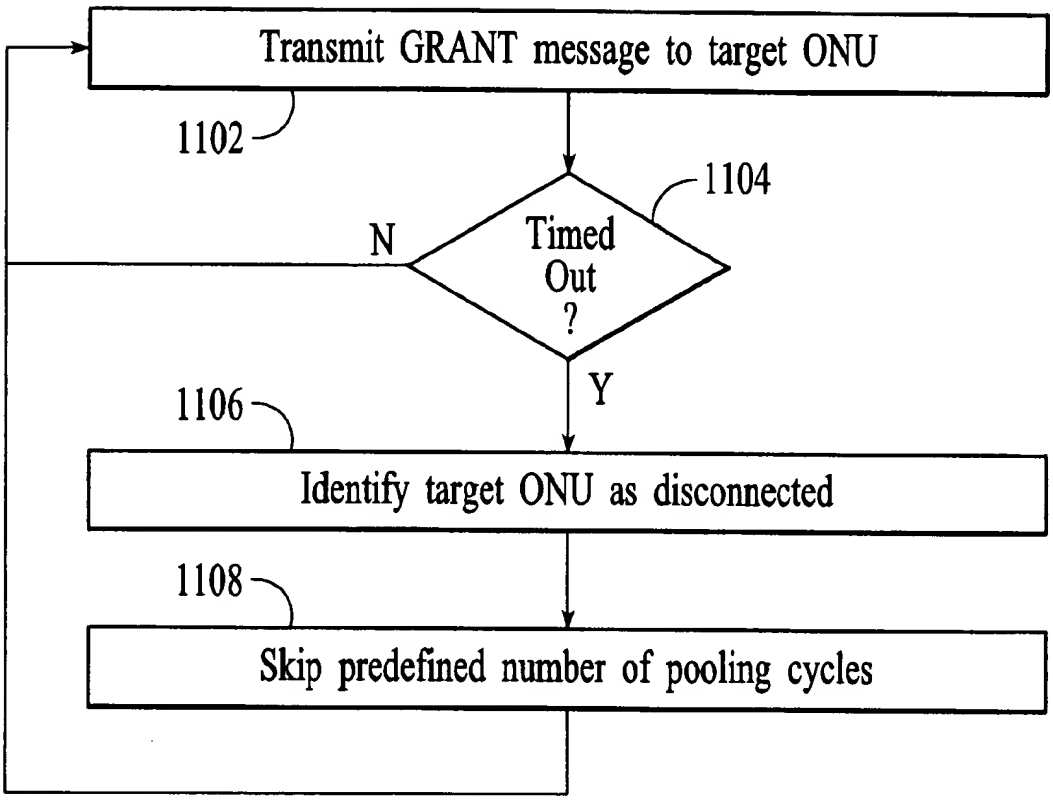

Transmit GRANT message to target ONU
1102
1104
Timed Out ?
N
Y
1106
Identify target ONU as disconnected
1108
Skip predefined number of pooling cycles

FIG. 11

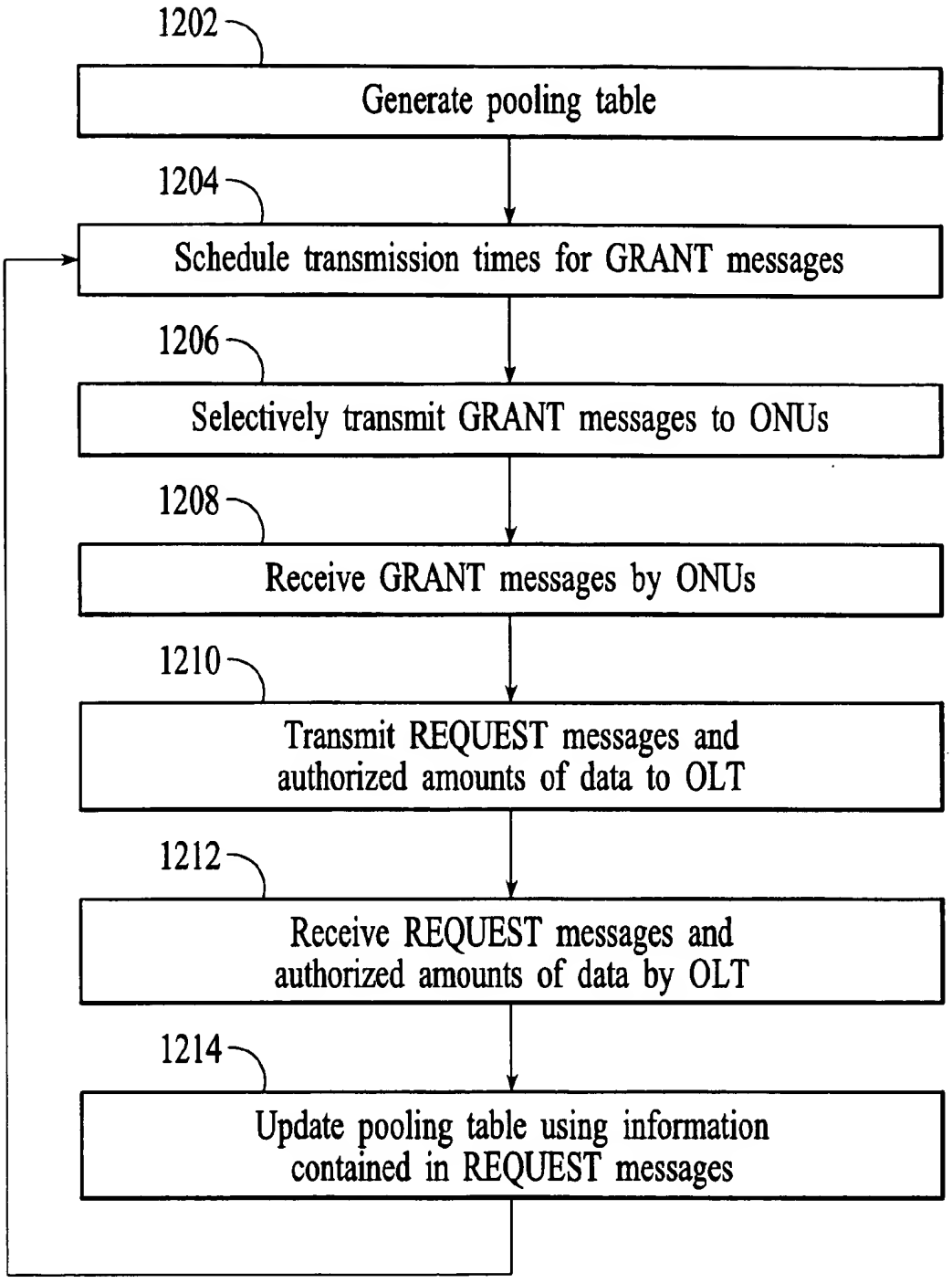

1202
Generate pooling table
1204
Schedule transmission times for GRANT messages
1206
Selectively transmit GRANT messages to ONUs
1208
Receive GRANT messages by ONUs
1210
Transmit REQUEST messages and authorized amounts of data to OLT
1212
Receive REQUEST messages and authorized amounts of data by OLT
1214
Update pooling table using information contained in REQUEST messages

FIG. 12

METHOD AND SYSTEM FOR DYNAMIC BANDWIDTH ALLOCATION IN AN OPTICAL ACCESS NETWORK

FIELD OF THE INVENTION

The invention relates generally to access networks, and more particularly to a passive optical access network.

BACKGROUND OF THE INVENTION

The explosion of the Internet and the desire to provide multiple communications and entertainment services to end users have created a need for a broadband network architecture that improves access to end users. Although the bandwidth of backbone networks has experienced a substantial growth in recent years, the bandwidth provided by access networks has remained relatively unchanged. Thus, the "last mile" still remains a bottleneck between a high capacity LAN or Home network and the backbone network infrastructure.

Digital Subscriber Line (DSL) and Cable Modem (CM) technologies offer some improvements over more conventional last mile solutions. However, these technologies still do not provide enough bandwidth to support emerging services such as Video-On-Demand (VoD) or two-way video conferencing. In addition, not all customers can be covered by DSL and CM technologies due to distance limitations.

One broadband access network architecture that offers a solution to the "last mile" problem is a point-to-multipoint passive optical network (PON). A point-to-multipoint PON is an optical access network architecture that facilitates broadband communications between an optical line terminal (OLT) and multiple remote optical network units (ONUs) over a purely passive optical distribution network. A point-to-multipoint PON utilizes passive fiber optic splitters and combiners to passively distribute optical signals between the OLT and the remote ONUs.

FIGS. **1A** and **1B** illustrate the management of network traffic in a point-to-multipoint PON. As an example, the PON is shown to include an OLT **102** and three ONUs **104**, **106** and **108**, although the PON may include additional ONUs. Referring to FIG. **1A**, the OLT includes an optical transmitter **110** that sends downstream traffic containing ONU-specific information blocks **1**, **2** and **3** to the ONUs. The downstream traffic is optically broadcasted by a passive optical splitter/combiner **112** into three separate signals that each carries all of the ONU-specific information blocks. The ONUs **104**, **106** and **108** include optical receivers **114**, **116** and **118**, respectively, that receive all the information blocks transmitted by the OLT. Each ONU then processes the information blocks that are intended for that ONU and discards the information blocks that are intended for the other ONUs. For example, ONU-**1** receives information blocks **1**, **2**, and **3**. However, ONU-**1** only delivers information block **1** to end user **1**. Likewise, ONU-**2** only delivers information block **2** to end user **2** and ONU-**3** only delivers information block **3** to end user **3**.

Referring to FIG. **1B**, the ONUs **104**, **106** and **108** also include optical transmitters **120**, **122** and **124**, respectively, to transmit upstream traffic to OLT **102**. The upstream traffic is managed utilizing a time division multiplex access (TDMA) protocol, in which specific transmission time slots are dedicated to individual ONUs. The ONU-specific time slots are synchronized so that upstream information blocks from the ONUs do not interfere with each other once the information blocks are combined onto the common fiber.

For example, ONU-**1** transmits information block **1** in a first ONU-specific time slot, ONU-**2** transmits information block **2** in a second ONU-specific time slot, and ONU-**3** transmits information block **3** in a third ONU-specific time slot. The time division multiplexed upstream traffic is then received by an optical receiver **126** of the OLT.

A concern with a TDMA PON is that the bandwidth of the PON is not always efficiently utilized for data transmission because network traffic typically exhibits a high degree of burstiness. The bursty network traffic results in some transmission time slots that consistently overflow even under a very light traffic load. In addition, the bursty network traffic results in some transmission time slots that are not completely filled even when the overall traffic load is very high.

In view of the above concern, there is a need for an access network based on PON technology that efficiently utilizes the available bandwidth by reducing the amount of bandwidth wasted because transmission time slots are not filled to the maximum capacity.

SUMMARY OF THE INVENTION

An optical access network and method for transmitting optical data in the network utilizes an interleaved polling scheme to efficiently use the available bandwidth of the network. The use of the interleaved polling scheme allows a central terminal of the network to dynamically allocate upstream bandwidth from remote terminals of the network to the central terminal in response to the amount of data that is waiting at the remote terminals to be transmitted to the OLT. In one embodiment, the optical access network is based on Passive Optical Network (PON) technology. In another embodiment, the optical access network utilizes Ethernet protocol to encapsulate data in Ethernet frames for transmission. Thus, in these embodiments the optical access network includes all of the advantages associated with the PON technology and/or the Ethernet protocol. In addition, since the allocation of upstream bandwidth is on an as needed basis, loss of bandwidth due to unfilled time slots is substantially eliminated.

A method of transmitting optical data in an optical network in accordance with the invention includes the steps of generating a table that includes information about the current sizes of data waiting to be transmitted from a plurality of remote terminals to a central terminal, selectively transmitting grant messages to the remote terminals, receiving authorized amounts of the data from the remote terminals and request messages containing updated information about the current sizes of the data waiting to be transmitted at the remote terminals in response to the grant messages, and updating the table using the updated information contained in the request messages received from the remote terminals. Each grant message is indicative of a permission for a targeted remote terminal to transmit an authorized amount of data waiting at the targeted remote terminal, which is dependent on the information included in the table with regard to the targeted remote terminal. In an embodiment, the optical network is a passive optical network (PON), which may be an Ethernet-based PON.

In an embodiment, the table that-includes information about the current sizes of data waiting to be transmitted from the plurality of remote terminals further includes information about round trip times of data transmission between the central terminal and the remote terminals. In this embodiment, the method may include steps of computing current round trip times for the remote terminals, including monitoring transmission times associated with the grant

messages from the central terminal and reception times associated with the authorized amounts of the data from the remote terminals, and updating the information about round trip times of data transmission using the computed current round trip times.

In an embodiment, the step of updating the table includes subtracting a value from a new entry for the targeted remote terminal. The value corresponds to the actual amount of data transmitted from said targeted remote terminal, while the new entry corresponds to the updated information contained in a request message from the targeted remote terminal.

In an embodiment, the method may further include a step of scheduling transmission times for the grant messages such that the data and the updated information from the remote terminals do not overlap during transmission. The transmission times for the grant messages substantially define receptions times for the request messages at the central terminal. The step of scheduling transmission times may include rescheduling an original transmission time for a grant message to a rescheduled transmission time when the original transmission time conflicts with anther transmission time for a different grant message.

In an embodiment, the method further includes steps of detecting a disconnected remote terminal, and reducing the frequency of grant messages that are transmitted to the disconnected remote terminal. The step of detecting a disconnected remote terminal may include waiting a predefined period for the disconnected remote terminal to respond to a grant message transmitted to the disconnected remote terminal.

An optical access network in accordance with the invention includes a plurality of remote terminals and a central terminal. In one embodiment, the optical access network is an Ethernet-based PON. The remote terminals of the network receive and transmit optical data. Each remote terminal is configured to transmit a request message and an authorized amount of data waiting at the remote terminal in response to a grant message received by the remote terminal. The request message includes updated information about the current size of data waiting at the remote terminal, while the grant message includes information that indicates the authorized amount. The central terminal of the network is optically coupled to the remote terminals to transmit and receive the optical data.

The central terminal includes memory that has a table containing latest information about the current sizes of data waiting to be transmitted from the remote terminals to the central terminal, and a processor that is configured to selectively transmit grant messages to the remote terminals using the latest information contained in the table to indicate authorized amounts of data that can be sent by receiving remote terminals. The processor of the central terminal is further configured to receive request messages from the remote terminals in response to the grant messages and to update the table in memory using the updated information contained in the request messages.

In an embodiment, the processor of the central terminal is configured to update the table in memory by subtracting values from new entries for selected remote terminals that have transmitted the request messages to the central terminal. The values correspond to the actual amounts of data transmitted from the selected remote terminals, while the new entries correspond to the updated information contained in the request messages.

In an embodiment, the processor of the central terminal is further configured to schedule transmission times for the grant messages such that the data and the request messages from the remote terminals do not overlap during transmission. In this embodiment, the processor may be further configured to reschedule an original transmission time for a specific grant message to a rescheduled transmission time when the original transmission time conflicts with another transmission time for a different grant message.

In an embodiment, the processor of the central terminal is configured to detect disconnected remote terminals by identifying the remote terminals that have not responded to the grant messages within a timeout period. In this embodiment, the processor may be configured to reduce the frequency of grant messages that are transmitted to the disconnected remote terminals.

In an embodiment, the table in memory further includes information about round trip times of data transmission between the central terminal and the remote terminals. In this embodiment, the processor of the central terminal may be configured to compute current round trip times for the remote terminals by monitoring transmission times associated with grant messages from the central terminal and reception times associated with request messages from the remote terminals and to update the information about round trip times using the current round trip times.

Other aspects and advantages of the present invention will become apparent from the following detailed description, taken in conjunction with the accompanying drawings, illustrated by way of example of the principles of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1A** illustrates the downstream flow of traffic from an OLT to multiple ONUs in a point-to-multipoint PON.

FIG. **1B** illustrates the upstream flow of traffic from the ONUs to the OLT in the point-to-multipoint PON.

FIG. **2** is a block diagram of an optical access network based on PON technology in accordance with the present invention.

FIGS. **3–8** illustrate the polling scheme of the optical access network.

FIG. **9** illustrates the embedding of a GRANT message in an Ethernet frame by the OLT using an escape code.

FIG. **10** is a flow diagram of an operation of the OLT with respect to GRANT message scheduling conflicts.

FIG. **11** is a process flow diagram of a detecting and polling procedure of the OLT with respect to disconnected ONUs.

FIG. **12** is a process flow diagram of a method of transmitting optical data in the optical access network in accordance with the present invention.

DETAILED DESCRIPTION OF THE INVENTION

With reference to FIG. **2**, an optical access network **202** that utilizes an interleaved polling scheme in accordance to the present invention is shown. The optical access network is based on Passive Optical Network (PON) technology. Therefore, the optical access network embodies the advantages associated with a PON. Furthermore, the interleaved polling scheme of the optical access network efficiently utilizes the available bandwidth of the network by allocating bandwidth on an as needed basis.

In an exemplary embodiment, the optical access network **202** utilizes the IEEE 802.3 protocol (commonly referred to

as Ethernet) to encapsulate data in Ethernet frames for transmission. Although the optical access network **202** may utilize other protocols, such as ATM data link protocol, the use of Ethernet protocol is preferred. Currently, about ninety-five percent (95%) of LANs use the Ethernet protocol. Therefore, an Ethernet-based PON is much more preferable than an ATM-based PON to interconnect Ethernet networks. In addition, ATM imposes a cell tax on variable length IP packets, since an IP packet that is longer than 48 bytes must be transmitted in two or more ATM cells. As the number of ATM cells increase, the bandwidth consumed by header information increases accordingly. Furthermore, ATM equipment is more expensive than Ethernet equipment.

The optical access network **202** includes an optical line terminal (OLT) **204**, which can be considered as a central terminal of the network, and a plurality of optical network units (ONUs) **206**, **208** and **210**, which can be considered as remote terminals of the network. For illustrative purposes, the optical access network is described as including only three ONUs, although the network may include additional ONUs. Furthermore, although the optical access network is shown as being configured in a tree topology, the optical access network may be configured in other network topologies, such as a ring topology or a bus topology. The topology of the optical access network is not critical to the invention.

Each ONU **206**, **208** or **210** of the optical access network **202** includes a buffer **212**, an optical transceiver **214** and an ONU processor **216**. Each buffer of the ONUs operates as a temporary storage medium to store bytes of data that are to be transmitted to the OLT **204**. The bytes of data stored in the buffer include data from end users (not shown) supported by the respective ONU. Each optical transceiver of the ONUs operates to transmit some or all of the bytes of data stored in the buffer as Ethernet frames in the form of optical signals to the OLT. In addition, each optical transceiver operates to receive optical signals from the OLT. Each ONU processor of the ONUs performs various operations to ensure that the respective ONU is properly functioning. Furthermore, each ONU processor executes steps in accordance with the interleaved polling scheme of the optical access network, which is described in detail below.

The OLT **204** of the optical access network **202** includes an optical transceiver **218**, an OLT memory **220** and an OLT processor **222**. The optical transceiver of the OLT is similar to the optical transceivers **214** of the ONUs **206**, **208** and **210**. The OLT optical transceiver operates to transmit optical signals to the ONUs, and to receive optical signals from the ONUs. The OLT memory **220** stores a polling table **224** to facilitate the interleaved polling scheme of the optical network access. The OLT memory may be any type of storage medium, such as DRAM or FPGA. Although the OLT memory is shown in FIG. **2** as a separate component from the OLT processor, the OLT memory may instead be incorporated in the OLT processor. The polling table is illustrated as having exemplary data for a particular instant. The polling table includes information regarding the ONUs supported by the OLT. This ONU information is illustrated in the, column of the polling table under the heading "ONU". The polling table also includes information about the number of bytes of data buffered at each of the ONUs waiting to be transmitted to the OLT. The buffered data information is illustrated in the column of the polling table under the heading "Bytes". Furthermore, the polling table includes information about the round-trip times (RTTs) for data to travel between the OLT and the ONUs. The RTT information is illustrated in the column of the polling table under the heading "RTT". An RTT value included in the polling table includes the actual RTT as well as processing times required to process and generate control messages. The polling table stored in the memory of the OLT is used by the OLT processor to efficiently regulate the network traffic between the OLT and the ONUs using the interleaved polling scheme.

The optical access network **202** also includes at least one optical splitter/combiner **226** and optical fibers **228**, **230**, **232** and **234** that provide connections between the OLT **204** and the ONUs **206**, **208** and **210** through the optical splitter/combiner. The optical fiber **228** connects the OLT to the optical splitter/combiner. The optical fibers **230**, **232** and **234** connect the optical splitter combiner to the ONUs **206**, **208** and **210**, respectively. The optical fibers **228**, **230**, **232** and **234** may be used for both downstream (OLT to ONUs) and upstream (ONUs to OLT) data transmissions. In an alternative configuration, these optical fibers may be used for only one-way data transmissions. In such a configuration, the optical access network includes dual optical fibers to allow two-way data transmissions.

The operation of the optical access network **202** with respect to downstream traffic management is similar to the operation of a traditional Ethernet network, which by design broadcast data throughout the Ethernet network. In the downstream direction, optical Ethernet frames are broadcast by the OLT **204** and then extracted by their destination ONU based on the media-access control (MAC) addresses of the frames. Therefore, in the downstream direction, there is no issue of competing bandwidth usages since the OLT is the only data-transmitting source. Consequently, the total available bandwidth is almost fully utilized in the downstream. However, in the upstream direction, ONUs must share the channel capacity of the optical fiber **228** and other resources. The upstream bandwidth of the optical access network is clearly limited as the sum of peak ingress rates R_D of all ONUs **206**, **208** and **210** exceeds the PON throughput R_U, or the upstream channel capacity. Furthermore, since network traffic typically exhibits a high degree of burstiness, a time-division multiple access (TDMA) system with fixed time slots cannot fully utilize the upstream bandwidth. In TDMA systems, the bursty network traffic results in some transmission time slots that consistently overflow even under a very light traffic load. In addition, the bursty traffic results in some transmission time slots that are not completely filled even when the overall traffic load is very high.

In contrast to the TDMA systems with fixed time slots, the optical access network **202** maximizes the upstream bandwidth utilization by using an interleaved polling scheme to manage upstream traffic. The polling scheme of the optical access network is described with reference to FIGS. **3–8** using specific values as examples, such as the byte sizes of data in buffers of the ONUs and the RTT values for each ONU.

For a given moment of time (t=0) after the initialization of the OLT **204**, a polling table **302** stored in the memory **220** of the OLT is assumed to be current, as illustrated in FIG. **3**. That is, the polling table includes updated information about the active ONUs, the number of bytes waiting at each ONU's buffer, and the round-trip time (RTT) for each ONU. At t=0, the current polling table includes the following information. For the ONU-1, there are 1200 bytes of data waiting in the buffer **212** and the RTT is 200 μs. For the ONU-2, there are 400 bytes of data waiting in the buffer **212** and the RTT is 170 μs. For the ONU-3, there are 2500 bytes of data waiting in the buffer **212** and the RTT is 120 μs.

As illustrated in FIG. **3**, at time t=0, the OLT **204** transmits an ONU control message **304** to the ONU-1, which autho-

rizes the ONU-1 to send 1200 bytes of data that is currently in the buffer **212** of the ONU-1. Such a message will be referred to herein as a GRANT message. Since the OLT broadcasts data to all ONUs in the downstream direction, a GRANT message contains identification of so the destination ONU in a NID ("node identification") field **306**, as well as the size of the granted window in bytes in a WS ("window size") field **308**, as illustrated by the GRANT message **304**. The GRANT message **304** is then received by the ONU-1, at time t=t=t1. The time t1 is approximately half of the RTT for the ONU-1. By the time the GRANT message is received by the ONU-1, additional bytes of data may have been received by the buffer of the ONU-1. As an example, the ONU-1 may have received an additional 3100 bytes of data. Therefore, the total number of bytes in the buffer of the ONU-1 is 4300 (1200+3100) bytes of data.

Upon receiving the GRANT message **304**, the ONU-1 sends an OLT control message **402** to the OLT **204**, as illustrated in FIG. **4**. Such a message will be referred to herein as a REQUEST message. Similar to a GRANT message, a REQUEST message also includes an NID field **404** and a WS field **406**. The WS field includes the current bytes of data waiting in the buffer of a particular ONU when the REQUEST message was generated. Thus, the REQUEST message **402** will indicate that 4300 bytes of data are currently waiting in the buffer **212** of the ONU-1. The REQUEST message tells the OLT how many bytes of data are in the buffer of the ONU-1 at the moment when the REQUEST message was generated. The ONU-1 also sends data **408** stored in the buffer of the ONU-1 up to the size of the granted window. In this example, the ONU-1 sends 1200 bytes of the data from the buffer of the ONU-1.

Even before the OLT **204** receives the REQUEST message **402** from the ONU-1, the OLT knows the time t_1^{last} when the last bit of transmission from the ONU-1 will arrive at the OLT. From the RTT information in the current polling table **302**, the OLT knows when the first bit from ONU-1 will arrive at the OLT. The first bit will arrive exactly after the RTT for the ONU-1. In addition, the OLT knows how many bytes (bits) will arrive at the OLT, since the OLT has authorized the ONU-1 to send a specific number of bytes by the GRANT message **304** to the ONU-1. Thus, the time t_1^{last} can be derived as follows:

$$t_1^{last} = t_1^G + RTT_1 + \frac{WS_1}{\text{Line\_Rate}} =$$

$$0 + 200\ \mu s + \frac{1200\ \text{bytes} \times 8\ \frac{\text{bits}}{\text{byte}}}{10^9\ \frac{\text{bits}}{\text{sec}}} = 209.6\ \mu s$$

In the above equation, t_1^G is the time when the. GRANT message was sent to the ONU-1, RTT_1 is the RTT for ONU-1, WS_1 is the granted window size, and Line_Rate is the data transmission speed for the optical access network **202**.

Since the OLT.**204** knows when the last bit from the ONU-1 will arrive, the OLT can send a GRANT message **502** to the ONU-2 so that the first bit of transmission from the ONU-2 will arrive right after the last bit of transmission from the ONU-1, as illustrated in FIG. **5**. The time t_2^G for the OLT to send the GRANT message to the ONU-2 can be derived as follows:

$$t_2^G = t_1^{last} - RTT_2 + \text{Guard\_Band} = 209.6 - 170 + 5 = 44.6\ \mu s.$$

In the above equation, a guard band time, for example, of 5 μs, is added to ensure that data from the ONU-2 avoids collision with data from the ONU-1. In general, guard bands provide protection for fluctuations of RTTs and control message processing times of the ONUs.

At time t=t2, the data from the ONU-1 arrives at the OLT **204** in response to the GRANT message **304** from the OLT, as illustrated in FIG. **6**. The time t2 is approximately equal to the current RTT for the ONU-1. For this example, the time t2 equals 199 us. Since the REQUEST message **402** from the ONU-1 was sent first, this message is received by the OLT prior to the data **408** from the buffer **212** of the ONU-1. The information contained in the WS field of the REQUEST message **402**, i.e., the number of bytes in the buffer of the ONU-1 when the REQUEST message was sent, is used by the OLT to update the polling table. The NID of the REQUEST message **402** lets the OLT know which entry of the polling table to update. Thus, in this example, the updated entry for the ONU-1 can be computed as follows:

$$\text{updated\_entry} = \text{new\_entry} - \text{old\_entry} = 4300 - 1200 = 3100 \text{ bytes.}$$

Thus, the "Byte" entry for the ONU-1 is updated to 3100, as illustrated by an updated polling table **602** in FIG. **6**. However, there may be situations when the granted window size is not fully filled by the data from an ONU. For example, if one or more higher-priority data packets arrive at an ONU after a Request message is sent, the ONU may send the higher-priority data packets before the other buffered data. The higher-priority data packets, which can have variable lengths, may not exactly fill the granted window size. The same problem will occur if an ONU requests a widow size greater than a maximum transmission window size, which is discussed below. If packet fragmentation is not available, then the ONU will send less data, leaving the remaining amount of data in the buffer. Updating the polling table using the above formula will result in an underestimated value for the updated_entry. An alternative approach would be for the OLT to update the table after all the data from a given ONU is received using the following formula:

$$\text{updated\_entry} = \text{new entry} - \text{bytes\_received.}$$

In addition to updating the current buffered data for a given ONU, the OLT keeps track of times when the GRANT messages and the corresponding REQUESTs are received. The OLT uses this information to constantly update the RTT entries for the respective ONUs. For this example, the actual RTT for the ONU-1 is 199 us. Thus, the RTT entry for the ONU-1 in the polling table **302** is changed from 200 us to 199 us, as illustrated by the updated polling table **602**.

Turning now to the ONU-2, the OLT **204** calculates the time when the last bit from the ONU-2 will arrive at the OLT in the same manner as described above with respect to the ONU-1. Hence, the OLT will know when to send a GRANT message to the ONU-3 so that the data from ONU-3 will arrive at the OLT shortly after the data from the ONU-2. At time t=t3, the data from the ONU-2 arrives at the OLT, as illustrate in FIG. **7**. When the REQUEST message from the ONU-2 is received by the OLT, the information contained in the latest polling table **602** is updated for the ONU-2, as illustrated by an updated polling table **702**. In this example, the REQUEST message from the ONU-2 indicates that the buffer of the ONU-2 is empty, since the REQUEST message reported the same number of bytes as the actual number of bytes sent to the OLT.

In a similar fashion, the OLT **204** calculates the time when the last bit of transmission from the ONU-3 will arrive at the OLT and sends the next GRANT message to the QNU-1, which starts the new polling cycle. Meanwhile, a REQUEST

message and data from the ONU-3 are received by the OLT. The REQUEST message from the ONU-3 is used by the OLT to update the latest polling table **702** to an updated polling table **802**, as illustrated in FIG. **8**. When the ONU-2 is polled in the next cycle, the grant size of the GRANT message is zero bytes, since the latest "Bytes" entry for the ONU-2 is zero. In response to the GRANT message, the ONU-2 again sends a REQUEST message containing the last number of bytes waiting in the buffer of the ONU-2. However, since the grant size of the GRANT message is zero bytes, no data from the buffer of the ONU-2 is sent to the OLT. If the buffer of the ONU-2 receives additional data after the previous REQUEST message from the ONU-2 was sent, then the ONU-2 is authorized to send the additional data in the following polling cycle.

A concern with the polling scheme of the optical access network **202** is that if the OLT **204** authorizes each ONU to send the entire data stored in the buffer in one transmission, the ONUs with high data volume could monopolize the entire bandwidth. In order to avoid such monopolization, there can be a limit to the maximum transmission size for each transmission. Thus, the OLT will apply this maximum size limit when sending the GRANT messages. Consequently, every ONU will be authorized to send as many bytes of data in the buffer as requested in a previous cycle up to the maximum size limit. The maximum size limit may be fixed. As an example, the fixed maximum size limit may be based on a Service Level Agreement for each ONU. In an alternative approach, the maximum size limit may be dynamic. As an example, the dynamic maximum size limit may be based on the average network traffic load.

In the exemplary embodiment, the GRANT and REQUEST messages are embedded between Ethernet frames or in an Ethernet frame using an escape sequence. Under the Ethernet standard IEEE 802.3, Gigabit Ethernet uses 8-to-10 bit encoding, i.e., every byte is being encoded as 10 bits before being sent over the medium. However, not all of the 10-bit values are valid encoding of an 8-bit value. One or more of these "non-valid" codes can be chosen to represent an escape code. A GRANT message or a REQUEST message can be attached to an escape code. The escape code informs the receiving unit that a GRANT or REQUEST message is present. The embedding of a GRANT message by the OLT **204** is illustrated in FIG. **9**. An Ethernet frame **902** in transmission is shown in FIG. **9**. The end of byte C of the Ethernet frame coincides with a transmission time t^G to send a GRANT message **904**. The OLT then inserts an escape code **906** and the GRANT message **904** at the end of byte C of the Ethernet frame, as shown in an embedded Ethernet frame **908**. In a similar manner, a REQUEST message can be embedded between Ethernet frames or within an Ethernet frame by an ONU. The receiving unit will recognize the beginning of the control sequence by reading the escape code. The receiving unit can then extract the control message, e.g., three (3) bytes, that follow the escape code before passing the rest of the received data to a standard Ethernet receiver.

The GRANT messages transmitted by the OLT **204** are scheduled using the following formula:

$$G_j^{[i+1]} = \mathrm{MAX}\begin{cases} G_j^{[i]} + r^{[i]} - r^{[i+1]} + \dfrac{W_j^{[i]}}{R_u} + B \\ G_{j-1}^{[i+1]} + r^{[i+1]}, \end{cases} \quad (1)$$

where

$G_j^{[i]}$—time epoch when j^{th} Grant to i^{th} ONU is to be transmitted (note that $G_j^{[i]}+r^{[i]}$ is a time epoch when j^{th} Request from i^{th} ONU is received)

$r^{[i]}$—Round-Trip Time (RTT) for i^{th} ONU

$W_j^{[i]}$—j^{th} Window Size for i^{th} ONU

R_u—Transmission speed (bit rate)

B—Guard Band (in μs).

In the above formula, the superscript in brackets identifies an ONU as following:

$$[x]=(x \; MOD \; N)+1.$$

The top line in the formula (1) states that the GRANT message to $[i+1]^{th}$ ONU is scheduled such that the responding REQUEST message arrives after the end of the transmission window from i^{th} ONU and the guard band time. The bottom line in the formula (1) states that the GRANT message cannot be sent before the previous REQUEST message from the same ONU is received, i.e., the interval between successive GRANT messages to the same ONU is at least the RTT to that ONU. This requirement is necessary since the GRANT message needs the requested window size information contained in the previous REQUEST message.

From the formula (1), it can be seen that the GRANT message to the $[i+1]^{th}$ ONU should be sent before the GRANT message to the i^{th} ONU, if the following inequality is satisfied.

$$r^{[i+1]} > r^{[i]} + \frac{W_j^{[i]}}{R_u} + B. \quad (2)$$

Therefore, the order of GRANT messages is generally different from the order of REQUEST messages. The REQUEST messages and the data, if any, from the ONUs arrive at the OLT in the same order in every cycle. However, since the GRANT messages are being scheduled with regard to corresponding RTTs and granted window sizes, the order of the GRANT messages may be different in every cycle.

Scheduling a GRANT message to $[i+1]^{th}$ ONU ahead of a GRANT message to i^{th} ONU is not a problem, since the order of GRANT messages is determined in a cycle prior to the actual transmission of the GRANT messages. However, scheduling the GRANT messages using the formula (1) may result in a GRANT message scheduling conflict. The conflict occurs when the absolute difference between the left and right terms of the inequality (2) is less than the transmission time of the GRANT message.

The OLT **204** of the optical access network **202** is configured to resolve the GRANT message scheduling conflict by rescheduling the GRANT message currently being scheduled for transmission. The operation of the OLT with respect to GRANT message scheduling conflicts is described with reference to FIG. **10**. At step **1002**, one of the ONUs **206**, **208** and **210** is selected by the OLT to schedule a transmission time for a GRANT message. The ONU selection may be predefined by a fixed sequence or may be dynamically adjusted in accordance with prescribed network conditions. Next, at step **1004**, the transmission time for the GRANT message to the selected ONU is tentatively scheduled using the formula (1). At step **1006**, a determination is made whether the tentatively scheduled transmission time conflicts with an established transmission time period for a GRANT message to another ONU.

If there is a conflict, the process proceeds to step **1008**, where the tentatively scheduled transmission time is delayed until after the established transmission time period for the conflicting GRANT message. That is, the GRANT message transmission time to the selected ONU is tentatively rescheduled for a time when the conflicting GRANT message will have already been sent. After the GRANT message

to the selected ONU has been tentatively rescheduled, the process proceeds back to step **1006**, where another determination is made whether the tentatively rescheduled transmission time conflicts with an established transmission time period for a GRANT message to a different ONU. In this fashion, the transmission time of the GRANT message to the selected ONU can be tentatively rescheduled until there is no scheduling conflict.

However, if a determination is made at step **1006** that there is no scheduling conflict, the process proceeds to step **1010**, where the latest tentatively scheduled transmission time for the selected ONU is finalized as the scheduled transmission time for the GRANT message to the selected ONU. The process then proceeds back to step **1002**, where the next ONU is selected to schedule a transmission time for a GRANT message to that ONU.

A consequence of using such a resolution for GRANT message scheduling conflicts is described using an example. In this example, the current GRANT message $G_j^{[h]}$ conflicts with the GRANT message $G_i^{[k]}$, which has already been scheduled. When the GRANT scheduling conflict is detected by the OLT **204**, the GRANT message $G_j^{[h]}$ will get scheduled after the time when lo the end of the GRANT message $G_i^{[k]}$ is transmitted. Assuming one Gigabit per second transmission speed, as an example, the transmission time for any GRANT message is equal to 32 ns. Therefore, the GRANT message $G_j^{[h]}$ will be delayed by at most 32 ns. The fact that the GRANT message $G_j^{[h]}$ is being scheduled after the GRANT message $G_i^{[k]}$ means that the data from the ONU-h in a cycle j is suppose to arrive after the data from the ONU-k in a cycle i, i.e., eitherj>i or both j=i and [h]>[k]. Thus, the delay of the GRANT message $G_j^{[h]}$ will result in a corresponding increase of a guard band. This guard band increase may be further increased if the delayed GRANT message $G_j^{[h]}$ conflicts with another GRANT message. In the extreme case, the GRANT message $G_j^{[h]}$ may conflict with at most N−1 GRANT messages, where N is the number of ONUs included in the optical access network **202**. For the case of N=16 and guard band time of 5 μs, the maximum acquired delay is 0.48 μs, which is less than ten percent (10%) of the guard band time. An advantage of the described scheduling conflict resolution is that the resolution does not introduce any scalability issues in terms of the value of N because an increase in the number of ONUs can only result in an increase of the guard band time.

Using the above-described polling scheme, the upstream channel of the OLT **204** is efficiently utilized to nearly one hundred percent (100%). However, some of the OLT upstream channel bandwidth is taken up by REQUEST messages that have to be embedded between data transmissions. In addition, the guard bands between the transmissions from different ONUs also take up some of the upstream channel bandwidth of the OLT. The guard bands include periods of time that are needed to readjust the sensitivity of the OLT receiver to receive optical signals from the ONUs that vary in distance from the OLT (burst mode receiving).

During the operation of the optical access network **202**, one or more ONUs may become disconnected due to either ONU failure or loss of power. Therefore, in one embodiment, the OLT **204** is designed to detect disconnected ONUs, and then to reduce the frequency of GRANT messages that are sent to the disconnected ONUs. The detecting and polling procedure of the OLT with respect to disconnected ONUs is described with reference to FIG. **11**. At step **1102**, a GRANT message is transmitted from the OLT to a target ONU. Next, at step **1104**, the OLT waits for a REQUEST message from the target ONU until a timeout interval has passed. If a REQUEST message from the target ONU is received within the timeout interval, the target ONU is not disconnected and thus, the process proceeds back to step **1102**, where another GRANT message is transmitted to the target ONU during the next polling cycle. However, if a REQUEST message from the target ONU is not received within the timeout interval, the target ONU is identified as being disconnected, at step **1106**. Next, at step **1108**, a predefined number of polling cycles are skipped for the disconnected ONU. As an example, the predefined number of polling cycles may be designed so that the disconnected ONU is only polled once per minute. The process then proceeds back to step **1102**, where another GRANT message is transmitted to the target ONU to determine if the target ONU is still disconnected. As illustrated in the process flow diagram of FIG. **11**, if the OLT receives a REQUEST message from an ONU previously marked as disconnected, then the ONU is then identified as active and is polled in the next polling cycle. This procedure can also be used for a cold start of the optical access network.

In this embodiment, GRANT messages are scheduled to be transmitted to the ONUs using the following formula:

$$G_j^{[i+1]} = \mathrm{MAX}\begin{cases} G_j^{[i]} + A^{[i]} - C^{[i+1]} + \dfrac{D_j^{[i]}}{R_u} + B \\ G_{j-1}^{[i+1]} + A^{[i+1]} \end{cases} \tag{3}$$

where

$$A^{[i]} = \begin{cases} \text{TIMEOUT}, & \text{if } ONU^{[i]} \text{ is disconnected} \\ r^{[i]}, & \text{otherwise} \end{cases}$$

$$C^{[i+1]} = \begin{cases} 0, & \text{if } ONU^{[i+1]} \text{ is disconnected} \\ r^{[i+1]}, & \text{otherwise} \end{cases}$$

$$D_j^{[i]} = \begin{cases} 0, & \text{if } ONU^{[i]} \text{ is disconnected} \\ W_j^{[i]}, & \text{otherwise.} \end{cases}$$

The above formula (3) is a modified version of the formula (1). For a disconnected ONU, the last known RTT value for that ONU cannot be used as a valid RTT value, since the reason for the disconnection could be, for example, relocation of that ONU. The formula (3) allows the OLT to schedule a GRANT message to a disconnected ONU with unknown R-TT. That is, the formula (3) allows the optical access network **202** to use interleaved schedule when the RTT values for some ONUs are unknown.

The modified formula (3) guarantees that the OLT **204** will not send a GRANT message to a disconnected ONU (with an unknown RTT) until the OLT receives the entire transmission from the previous ONU. Thus, data from a disconnected ONU with a very small RTT that suddenly comes alive will not collide with data from the previously polled ONU. In addition, the OLT will be able to schedule a GRANT message to the next ONU (i.e., keep the pipeline of the OLT full) even if the disconnected ONU remains disconnected. The reason is that since reply from a disconnected ONU can only be received within the timeout interval, the data from the next ONU can safely arrive after the timeout period, regardless of whether the disconnected ONU has come alive or remains disconnected.

An advantage of the optical access network **202** with the interleaved polling scheme is that there is no need to synchronize the ONUs. In addition, there is no need to

perform ranging (distance/delay equalization). Furthermore, since the entire scheduling and contention resolution are performed by the OLT **204**, the optical access network can easily change the scheduling in real-time based on changes in network conditions. There is no need for the ONUs to negotiate and agree on new parameters, nor do the ONUs need to switch to new settings synchronously because all the ONUs are driven by GRANT messages received from the OLT.

A method of transmitting optical signals in the optical network access **202** in accordance with the present invention is described with reference to FIG. **12**. At step **1202**, a polling table is generated by the processor **222** of the OLT **204** in memory **220** using the cold start procedure, as described above with reference to FIG. **11**. The polling table includes information about the current sizes of data waiting to be transmitted from the ONUs **206**, **208** and **210** to the OLT. At step **1204**, transmission times for GRANT messages to be sent are scheduled. Next, at step **1206**, the GRANT messages are selectively transmitted to the ONUs in accordance with the scheduled transmission times. Each GRANT message indicates an authorized amount of data that can be transmitted to the OLT from a receiving ONU. The authorized amount is dependent on the information included in the polling table. At step **1208**, the GRANT messages are received by the ONUs. In response to the GRANT messages, the receiving ONUs transmit REQUEST messages and the authorized amounts of data that were waiting for transmission at the receiving ONUs, at step **1210**. The REQUEST messages include updated information about the current sizes of data waiting to be transmitted at the receiving ONUs. Next, at step **1212**, the REQUEST messages and the authorized amounts of data are received by the OLT. At step **1214**, the polling table is updated by the processor of the OLT using the updated information contained in the REQUEST messages. After step **1214**, the method proceeds to repeat steps **1204–1214** using the updated polling table.

What is claimed is:

1. A method of transmitting data in an optical network comprising:

generating a table that includes information about the current sizes of data waiting to be transmitted from a plurality of remote terminals to a central terminal;

selectively transmitting grant messages to said remote terminals, each grant message being indicative of a permission for a targeted remote terminal to transmit an authorized amount of said data waiting at said targeted remote terminal, said authorized amount being dependent on said information included in said table with regard to said targeted remote terminal;

receiving authorized amounts of said data from said remote terminals in response to said grant messages, including request messages containing updated information about the current sizes of said data waiting to be transmitted at said remote terminals; and

updating said table using said updated information contained in said request messages received from said remote terminals.

2. The method of claim **1** wherein said step of updating said table includes subtracting a value from a new entry for said targeted remote terminal, said value corresponding to an actual amount of data transmitted from said targeted remote terminal, said new entry corresponding to said updated information contained in a request message from said targeted remote terminal.

3. The method of claim **1** further comprising a step of scheduling transmission times for said grant messages such that said data and said updated information from said remote terminals do not overlap during transmission, said transmission times for said grant messages substantially defining reception times for said request messages at said central terminal.

4. The method of claim **3** wherein said step of scheduling said transmission times includes scheduling said transmission times such that a grant message to a particular remote terminal is not transmitted before a previous request message from said particular remote terminal is received by said central terminal.

5. The method of claim **3** wherein said step of scheduling said transmission times includes rescheduling an original transmission time for a grant message to a rescheduled transmission time when said original transmission time conflicts with another transmission time for a different grant message.

6. The method of claim **5** wherein said step of rescheduling includes rescheduling said original transmission time for said grant message such that said rescheduled transmission time is after said different grant message has been transmitted.

7. The method of claim **1** further comprising:

detecting a disconnected remote terminal; and

reducing the frequency of grant messages that are transmitted to said disconnected remote terminal.

8. The method of claim **7** wherein said step of detecting said disconnected remote terminal includes waiting a predefined period for said disconnected remote terminal to respond to a grant message transmitted to said disconnected remote terminal.

9. The method of claim **1** further comprising a step of embedding said grant messages in between Ethernet frames or within said Ethernet frames, each grant message being embedded after a code of an Ethernet frame that is not used for Ethernet encoding, said code being utilized as an escape code.

10. The method of claim **1** wherein said authorized amount in said step of selectively transmitting said grant messages is less than a predefined maximum amount.

11. The method of claim **1** wherein said optical network is a passive optical network.

12. The method of claim **11** wherein said optical network is an Ethernet-based passive optical network.

13. The method of claim **1** wherein each of said grant messages and said request messages includes an identification code field and a window size field.

14. The method of claim **13** wherein said Window size field of each of said grant messages defines said authorized amount of data permitted to be transmitted by a receiving remote terminal.

15. The method of claim **13** wherein said window size field of each of said request messages defines the current size of data waiting to be transmitted to said central terminal at a transmitting remote terminal.

16. The method of claim **1** wherein said table further includes information about round trip times of data transmission between said central terminal and said remote terminals.

17. The method of claim **16** further comprising:

computing current round trip times for said remote terminals, including monitoring transmission times associated with said grant messages from said central terminal and reception times associated with said authorized amounts of said data from said remote terminals; and

updating said information about round trip times of data transmission using computed current round trip times.

18. A method of transmitting optical signals in a point-to-multipoint optical network comprising:

generating a table that contains entries of the current sizes of data waiting to be transmitted from a plurality of remote terminals to a central terminal;

transmitting a first grant message from said central terminal to a first remote terminal, said first grant message being indicative of a permission for said first remote terminal to transmit a specific amount of data waiting at said first remote terminal, said specific amount being dependent on an entry included in said table for said first remote terminal;

receiving said specific amount of said data from said first remote terminal at said central terminal in response to said first grant message, including a first request message containing updated information about the current size of said data waiting at said first remote terminal when said first request message was sent; and

updating said table using said updated information contained in said first request message received from said first remote terminal.

19. The method of claim 18 further comprising:

transmitting a second grant message from said central terminal to a second remote terminal;

receiving an authorized amount of said data from said second remote terminal at said central terminal, including a second request message containing updated information about the current size of said data waiting at said second remote terminal when said second request message was sent;

updating said table using said updated information contained in said second request message received from said second remote terminal.

20. The method of claim 19 further comprising a step of scheduling transmission times for said first and second grant messages such that said data and said first request message from said first remote terminal do not collide with said data and said second request message from said second remote terminal during transmission, said transmission times for said first and second grant messages substantially defining reception times for said first request message and said second request message at said central terminal.

21. The method of claim 20 wherein said step of scheduling said transmission times includes scheduling said transmission times such that said first grant message to said first remote terminal is not transmitted before a previous request message from said first remote terminal is received by said central terminal.

22. The method of claim 20 wherein said step of scheduling said transmission times includes rescheduling an original transmission time for said first grant message to a rescheduled transmission time when said original transmission time conflicts with another transmission time for a different grant message.

23. The method of claim 22 wherein said step of rescheduling includes rescheduling said original transmission time for said first grant message such that said rescheduled transmission time is after said different grant message has been transmitted.

24. The method of claim 18 further comprising;

detecting a disconnected remote terminal; and

reducing the frequency of grant messages that are transmitted to said disconnected remote terminal.

25. The method of claim 24 wherein said step of detecting said disconnected remote terminal includes waiting a predefined period for said disconnected remote terminal to respond to a grant message transmitted to said disconnected remote terminal.

26. The method of claim 18 wherein said step of updating said table includes subtracting a value from a new entry for said first remote terminal, said value corresponding to an actual amount of data transmitted from said first remote terminal, said new entry corresponding to said updated information contained in said first request message from said first remote terminal.

27. The method of claim 18 further comprising a step of embedding said first grant message in between Ethernet frames or within one of said Ethernet frames, said first grant message being embedded after a code of an Ethernet frame that is not used for Ethernet encoding, said code being utilized as an escape code.

28. The method of claim 18 wherein said specific amount in said step of transmitting said first grant message is less than a predefined maximum amount.

29. The method of claim 18 wherein said point-to-multipoint optical network is an Ethernet-based passive optical network.

30. The method of claim 18 wherein said table further includes information about round trip times of data transmission between said central terminal and said remote terminals.

31. The method of claim 30 further comprising:

computing a current round trip time for said first remote terminal, including monitoring a transmission time associated with said first grant message from said central terminal and a reception time associated with said specific amount of said data from said first remote terminal; and

updating said information about round trip times of data transmission for said first remote terminal using a computed current round trip time.

32. A point-to-multipoint optical network comprising:

a plurality of remote terminals that receive and transmit optical data, each remote terminal configured to transmit a request message and an authorized amount of data waiting at said remote terminal in response to a grant message received by said remote terminal, said request message including updated information about the current size of data waiting at said remote terminal, said grant message including information that indicates said authorized amount; and

a central terminal optically coupled to said remote terminals to transmit and receive said optical data, said central terminal comprising:

memory that includes a table containing latest information about the current sizes of data waiting to be transmitted from said remote terminals to said central terminal; and

a processor that is configured to selectively transmit grant messages to said remote terminals using said latest information contained in said table to indicate authorized amounts of data that can be sent by receiving remote terminals, said processor further configured to receive request messages from said remote terminals in response to said grant messages and to update said table in said memory using said updated information contained in said request messages.

33. The network of claim 32 wherein said processor of said central terminal is configured to update said table in said memory by subtracting values from new entries for selected remote terminals that have transmitted said request messages to said central terminal, said values corresponding to

actual amounts of data transmitted from said selected remote terminals, said new entries corresponding to said updated information contained in said request messages.

34. The network of claim **32** wherein said processor of said central terminal is further configured to schedule transmission times for said grant messages such that said data and said request messages from said remote terminals do not overlap during transmission, said transmission times for said grant messages substantially defining reception times for said request messages at said central terminal.

35. The network of claim **34** wherein said processor of said central terminal is further configured to schedule transmission times for said grant messages such that a specific grant message for a particular remote terminal is not transmitted before a previous request message from said particular remote terminal is received by said central terminal.

36. The network of claim **34** wherein said processor of said central terminal is further configured to reschedule an original transmission time for a specific grant message to a rescheduled transmission time when said original transmission time conflicts with another transmission time for a different grant message.

37. The network of claim **32** wherein said processor of said central terminal is configured to detect disconnected remote terminals by identifying said remote terminals that have not responded to said grant messages within a timeout period.

38. The network of claim **37** wherein said processor of said central terminal is configured to reduce the frequency of grant messages that are transmitted to said disconnected remote terminals.

39. The network of claim **32** wherein said processor of said central terminal is configured to embed said grant messages in between Ethernet frames or within said Ethernet frames, each grant message being embedded after a code of an Ethernet frame that is not used for Ethernet encoding, said code being utilizes as an escape code.

40. The network of claim. **32** wherein said processor of said central terminal is configured to generates said grant messages such that said authorized amounts of data that can be sent by said receiving remote terminals are less than a predefined maximum amount.

41. The network of claim **32** wherein said point-to-multipoint optical network is an Ethernet-based passive optical network.

42. The network of claim **32** wherein said table included in said memory further includes information about round trip times of data transmission between said central terminal and said remote terminals.

43. The network of claim **42** wherein said processor is configured to compute current round trip times for said remote terminals by monitoring transmission times associated with grant messages from said central terminal and reception times associated with request messages from said remote terminals and to update said information about round trip times using computed current round trip times.

* * * * *

US006118765A

United States Patent [19]

Phillips

[11] **Patent Number: 6,118,765**

[45] **Date of Patent: Sep. 12, 2000**

[54] **SYSTEM METHOD AND COMPUTER PROGRAM PRODUCT FOR ELIMINATING UNNECESSARY RETRANSMISSIONS**

[75] Inventor: **Marc S. Phillips**, San Diego, Calif.

[73] Assignee: **Qualcomm Inc.**, San Diego, Calif.

[21] Appl. No.: **09/006,685**

[22] Filed: **Jan. 13, 1998**

[51] **Int. Cl.**[7] **G06F 11/00**

[52] **U.S. Cl.** **370/235**; 370/252; 370/401; 370/428

[58] **Field of Search** 370/394, 389, 370/229, 230, 231, 235, 236, 252, 401, 428, 522; 714/749, 747, 748

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,442,637	8/1995	Nguyen	714/708
5,444,718	8/1995	Ejzak et al.	714/748
5,793,768	8/1998	Keshav	370/400
6,018,516	1/2000	Packer	370/231
6,031,818	2/2000	Lo et al.	370/216

OTHER PUBLICATIONS

"Auxiliary Timeout and Selective Packet Discard Schemes to Improve TCP Performance in PCN Environment" Byung-Gon Chun et al; IEEE 1997, pp. 381-385.

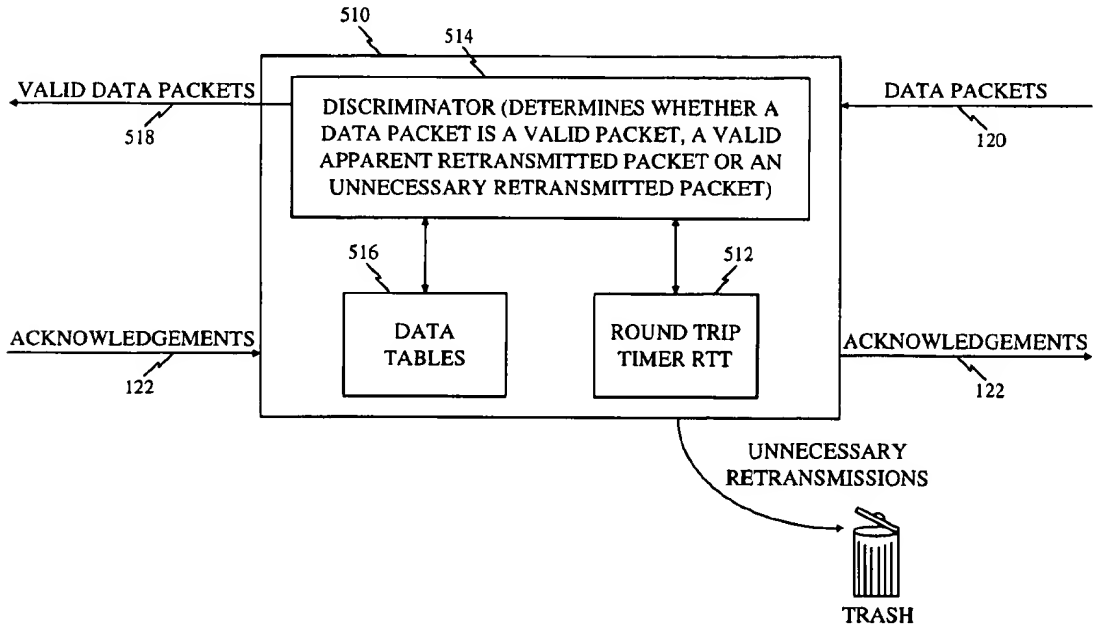
R. Braden, "Requirements for Internet Hosts—Communication Layers", Oct. 1989, pp. 1-108.

Primary Examiner—Douglas W. Olms
Assistant Examiner—Seema S. Rao
Attorney, Agent, or Firm—Philip Wadsworth; Thomas R. Rouse; Kent D. Baker

[57] **ABSTRACT**

A system, method and computer program product for preventing unnecessary retransmissions from being sent from a terminal such as, for example, an internet host computer, on a slow link. A slow link TCP optimizer receives data packets from a host computer and determines, based on an estimated round trip time, whether a data packet is a new data packet, a valid retransmitted data packet or an unnecessary retransmitted data packet. New data packets and valid retransmitted data packets are forwarded over the slow link to a remote terminal. Unnecessary retransmitted data packets are discarded. The estimating a round trip time is determined as the time it takes for a data packet to travel from the present invention to the remote terminal and for an acknowledgment of receipt to travel from the remote terminal back to the present invention. The estimated round trip time includes throughput latency that results from a slow link. The slow link TCP optimizer does not affect data packets that are sent from a host computer that properly accounts for a slow link when determining a round trip time.

19 Claims, 13 Drawing Sheets



SLOW LINK TCP OPTIMIZER

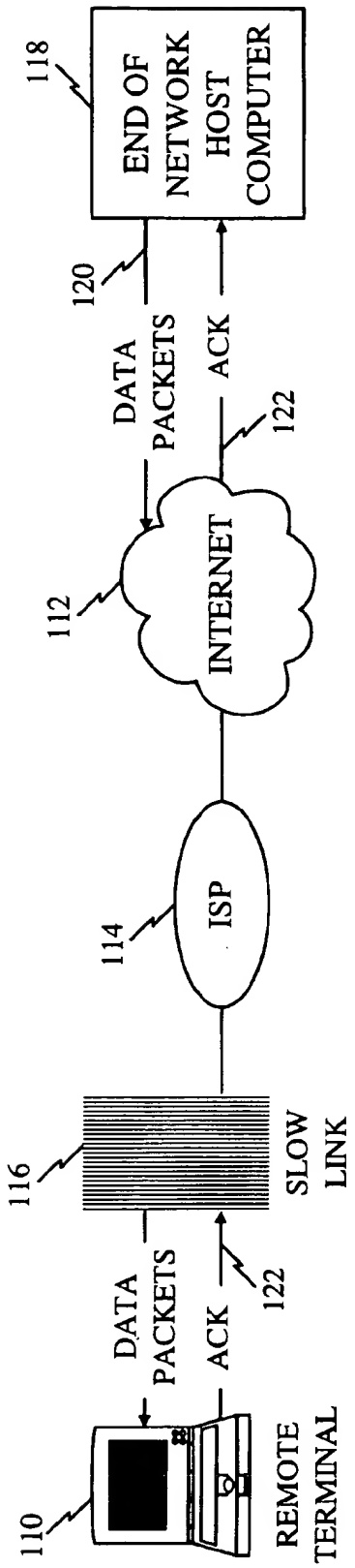

118
END OF NETWORK HOST COMPUTER
120
DATA PACKETS
ACK
122
112
INTERNET
114
ISP
116
SLOW LINK
DATA PACKETS
ACK
122
110
REMOTE TERMINAL

FIG. 1

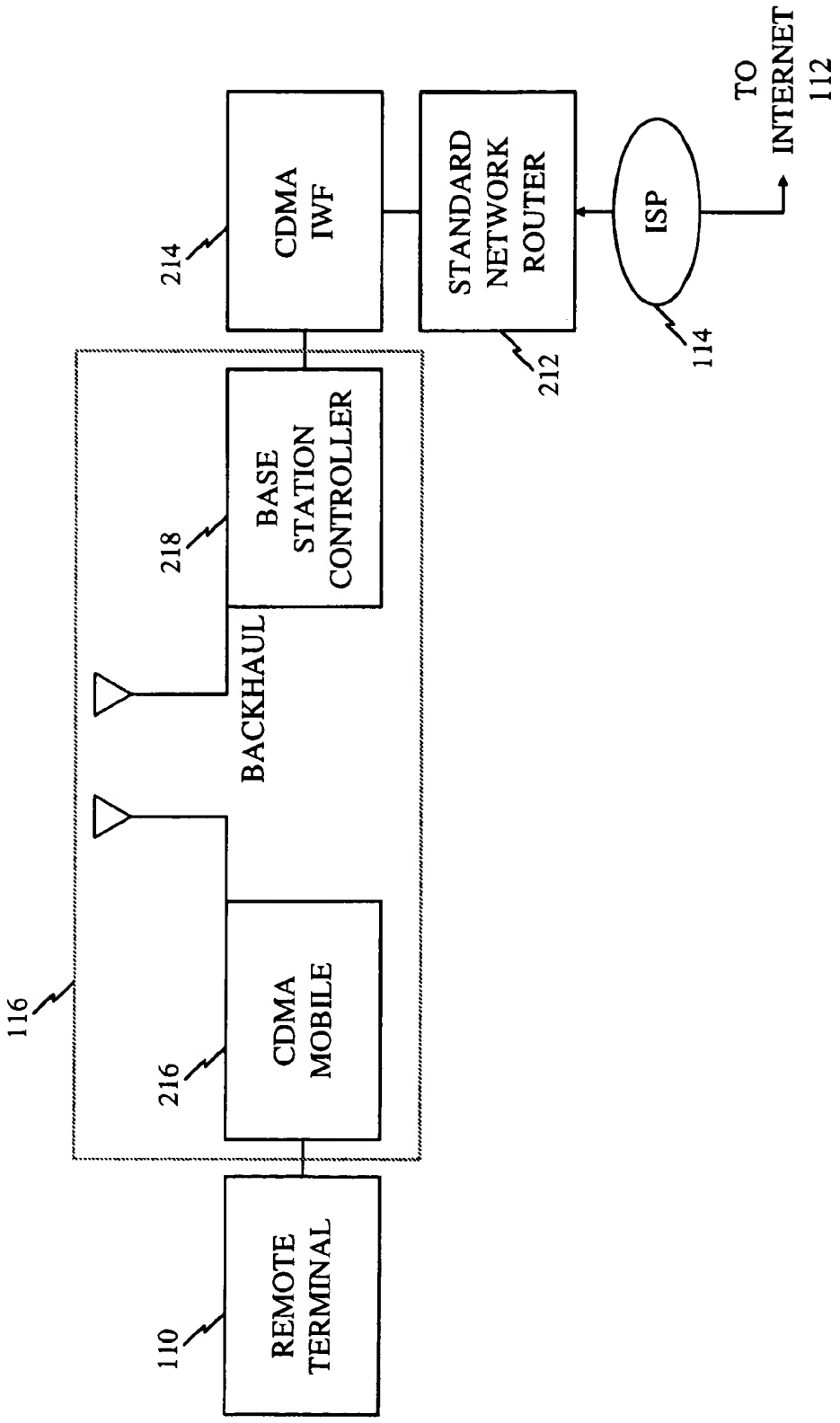

REMOTE TERMINAL
110
CDMA MOBILE
216
116
BACKHAUL
BASE STATION CONTROLLER
218
CDMA IWF
214
STANDARD NETWORK ROUTER
212
ISP
114
TO INTERNET
112

FIG. 2

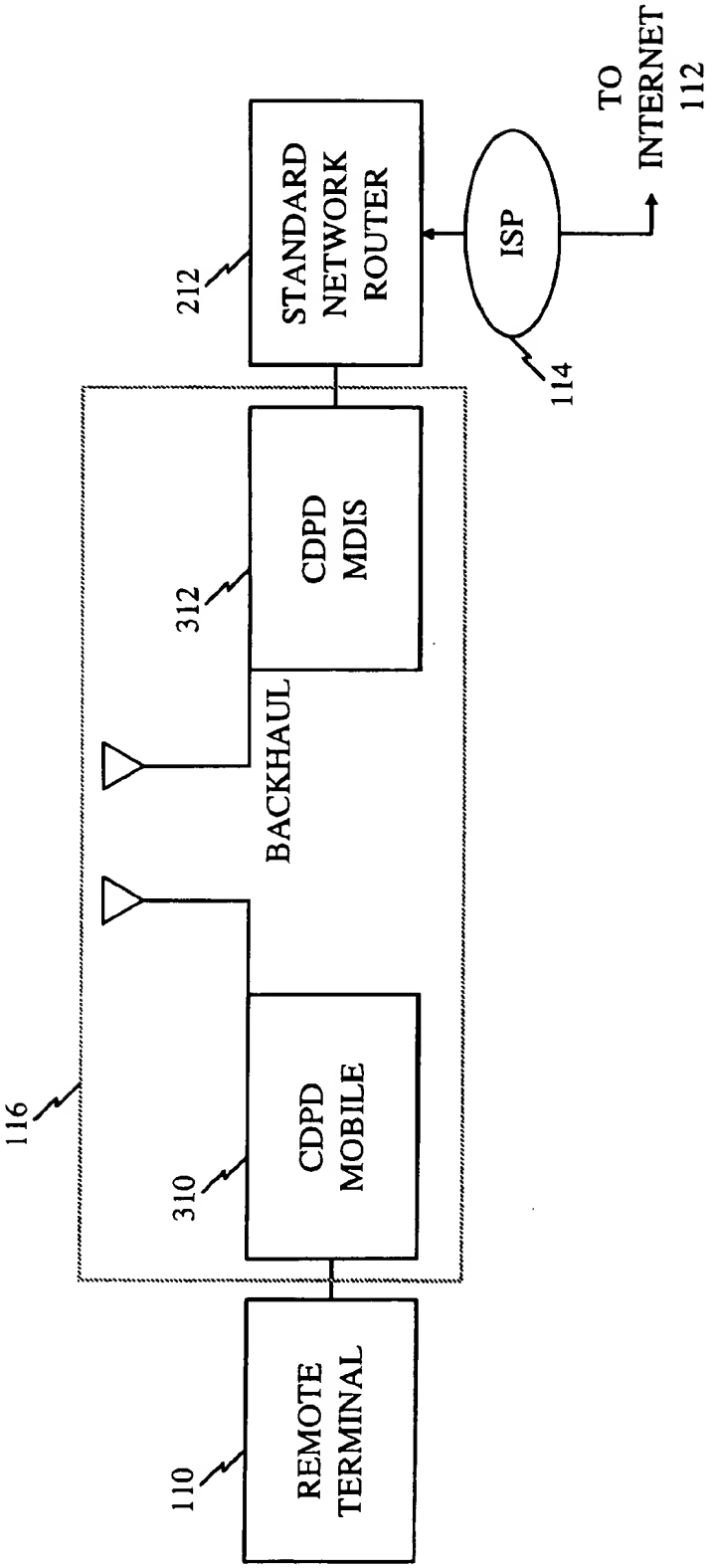

REMOTE TERMINAL
110
CDPD MOBILE
310
CDPD MDIS
312
BACKHAUL
STANDARD NETWORK ROUTER
212
ISP
114
TO INTERNET 112
116

FIG. 3

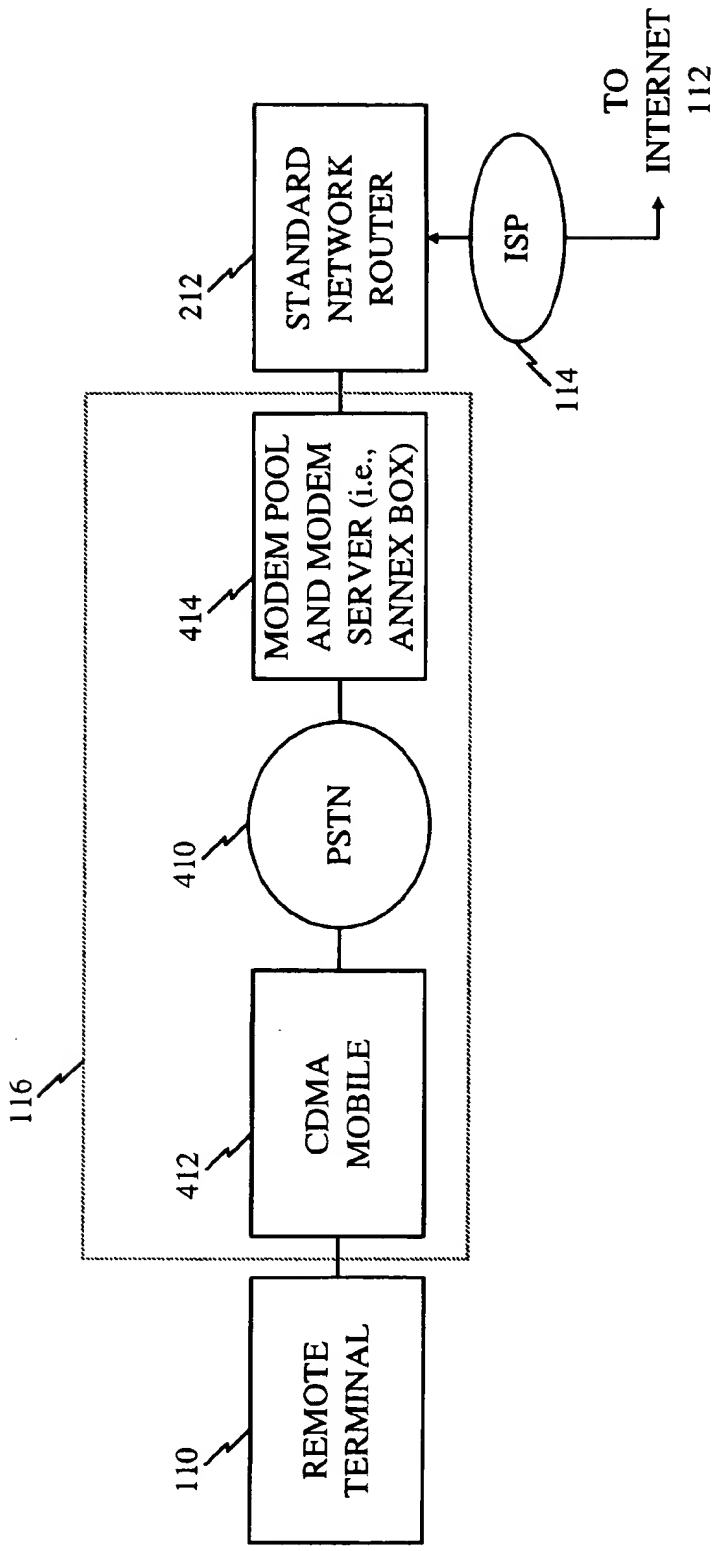

REMOTE TERMINAL
110
CDMA MOBILE
412
PSTN
410
MODEM POOL AND MODEM SERVER (i.e., ANNEX BOX)
414
116
STANDARD NETWORK ROUTER
212
ISP
114
TO INTERNET 112

FIG. 4

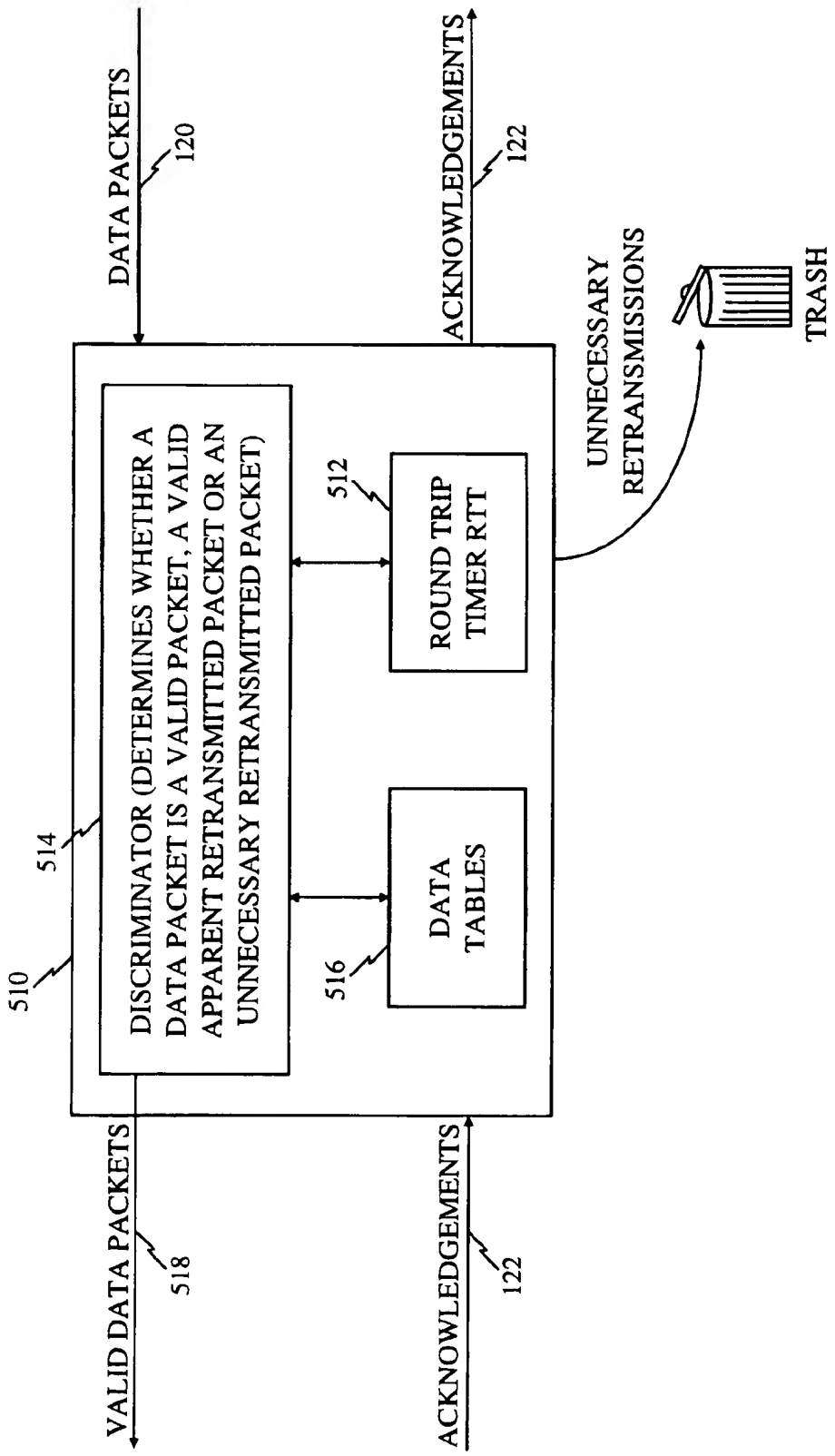

DATA PACKETS
120
ACKNOWLEDGEMENTS
122
UNNECESSARY RETRANSMISSIONS
TRASH
510
514
DISCRIMINATOR (DETERMINES WHETHER A DATA PACKET IS A VALID PACKET, A VALID APPARENT RETRANSMITTED PACKET OR AN UNNECESSARY RETRANSMITTED PACKET)
512
ROUND TRIP TIMER RTT
516
DATA TABLES
VALID DATA PACKETS
518
ACKNOWLEDGEMENTS
122
SLOW LINK TCP OPTIMIZER

FIG. 5

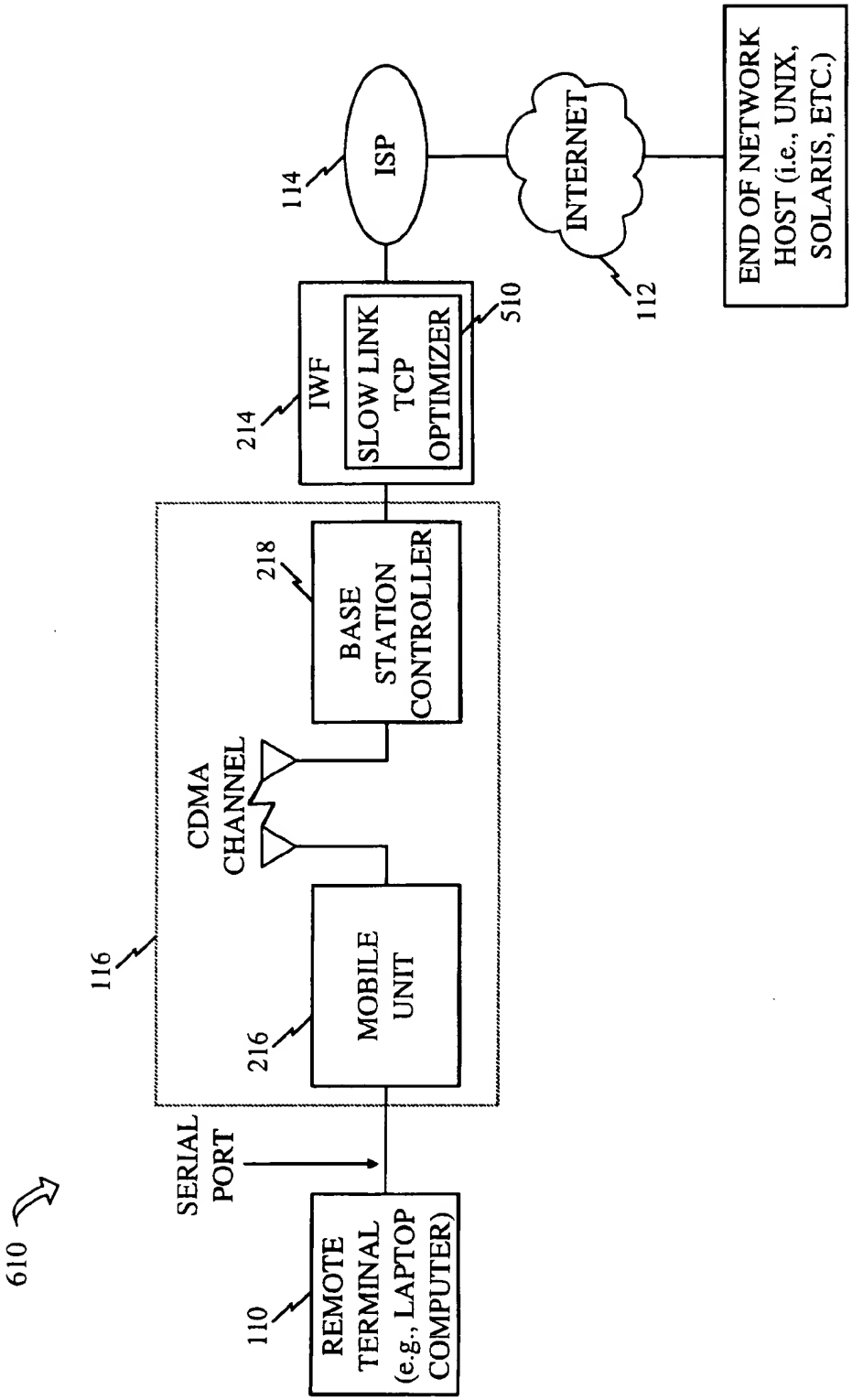

610
REMOTE TERMINAL (e.g., LAPTOP COMPUTER)
110
SERIAL PORT
116
MOBILE UNIT
216
CDMA CHANNEL
BASE STATION CONTROLLER
218
IWF
SLOW LINK TCP OPTIMIZER
214
510
ISP
114
INTERNET
112
END OF NETWORK HOST (i.e., UNIX, SOLARIS, ETC.)

FIG. 6

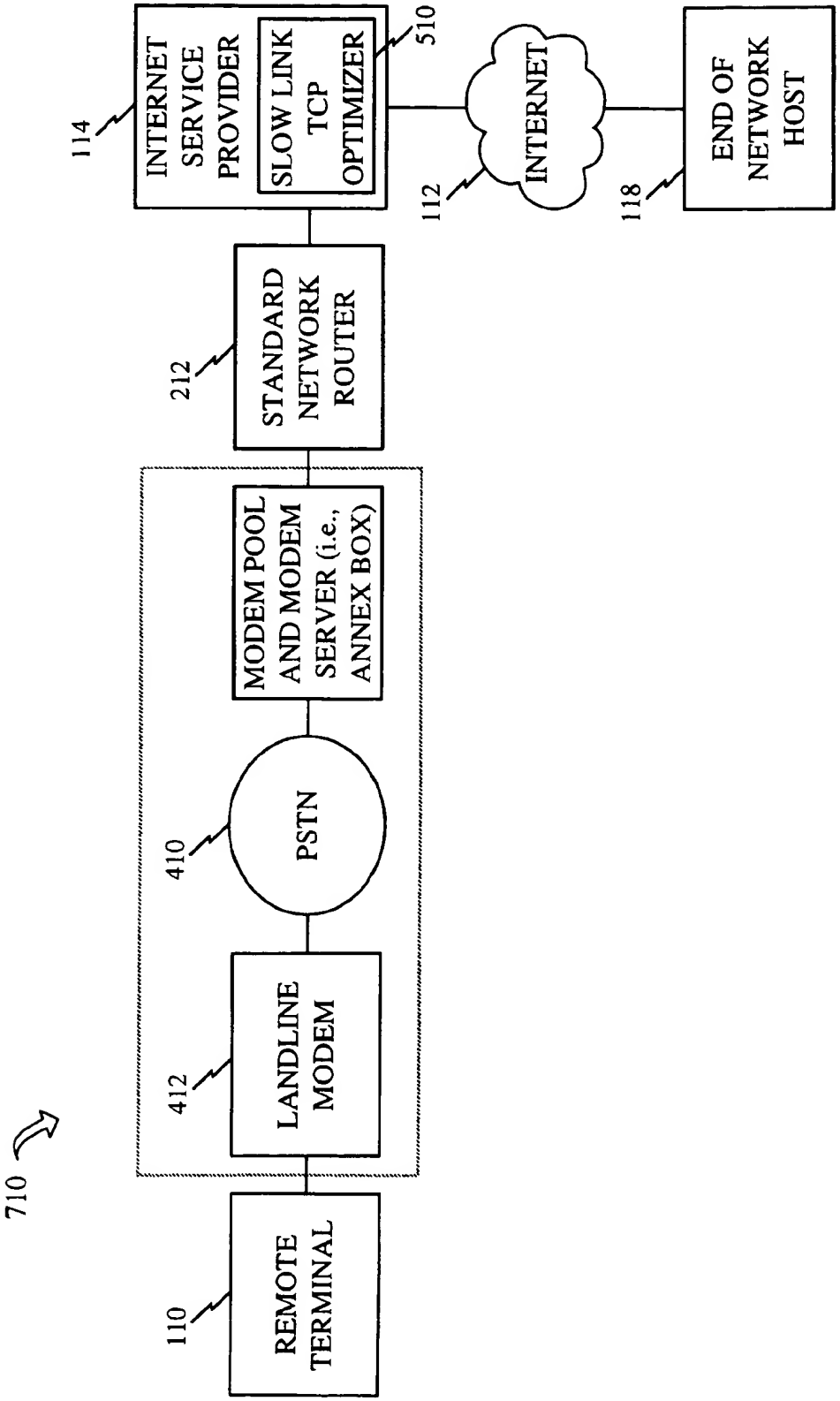

710
REMOTE TERMINAL
110
LANDLINE MODEM
412
PSTN
410
MODEM POOL AND MODEM SERVER (i.e., ANNEX BOX)
STANDARD NETWORK ROUTER
212
INTERNET SERVICE PROVIDER
SLOW LINK TCP OPTIMIZER
510
114
INTERNET
112
END OF NETWORK HOST
118

FIG. 7

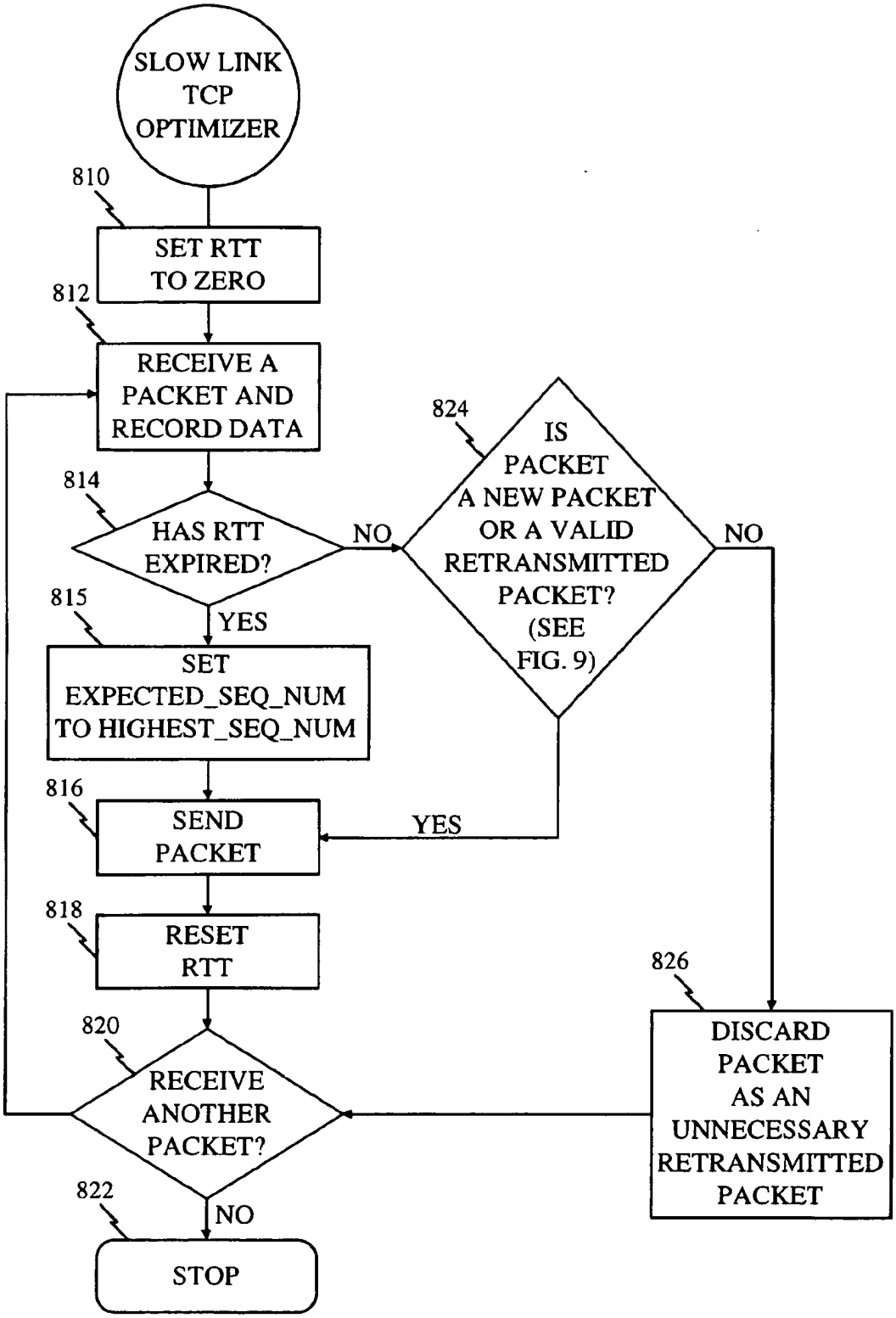
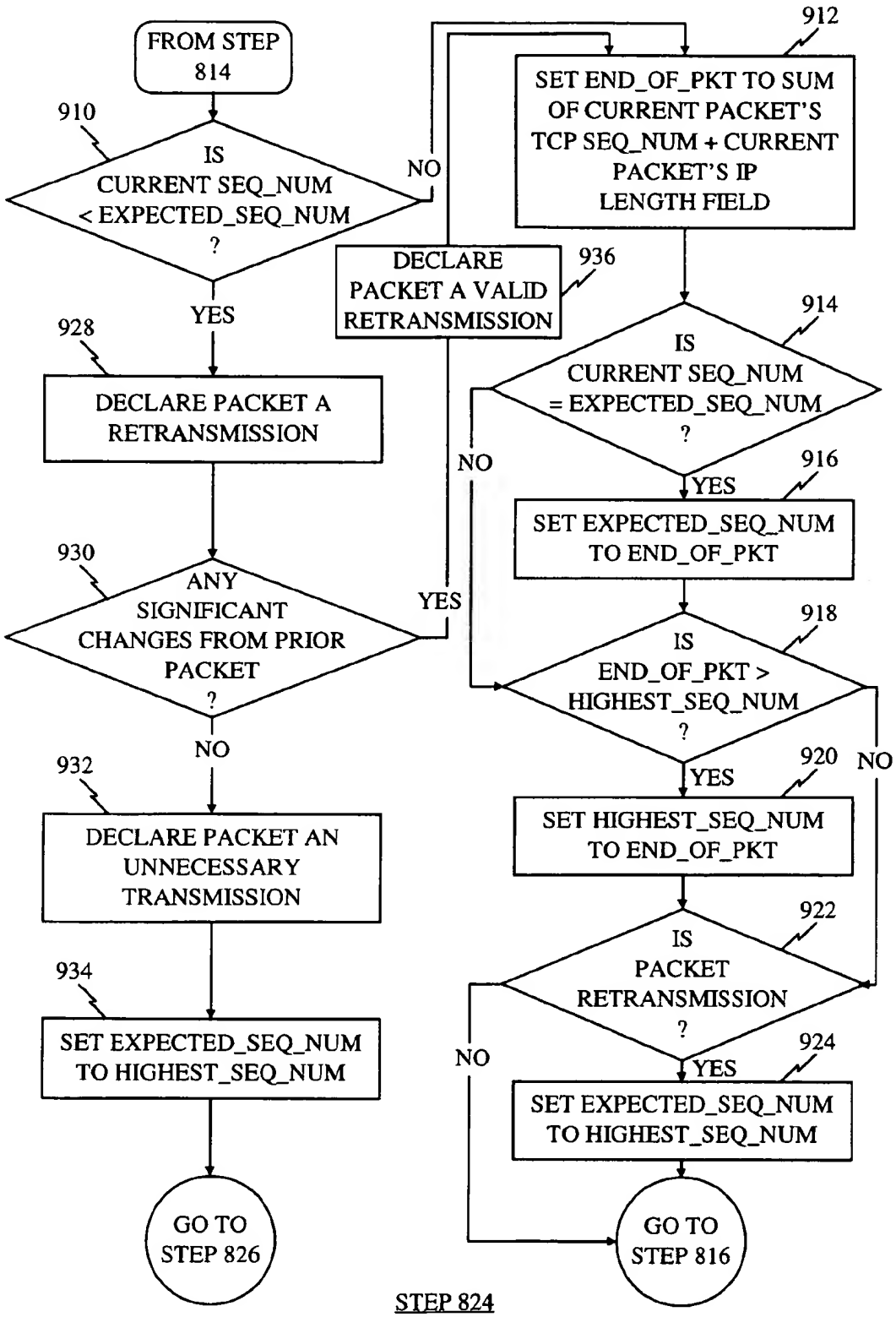

SLOW LINK TCP OPTIMIZER
810
SET RTT TO ZERO
812
RECEIVE A PACKET AND RECORD DATA
814
HAS RTT EXPIRED?
NO
824
IS PACKET A NEW PACKET OR A VALID RETRANSMITTED PACKET? (SEE FIG. 9)
NO
YES
815
SET EXPECTED_SEQ_NUM TO HIGHEST_SEQ_NUM
816
SEND PACKET
YES
818
RESET RTT
826
DISCARD PACKET AS AN UNNECESSARY RETRANSMITTED PACKET
820
RECEIVE ANOTHER PACKET?
822
NO
STOP

FIG. 8



STEP 824

FIG. 9

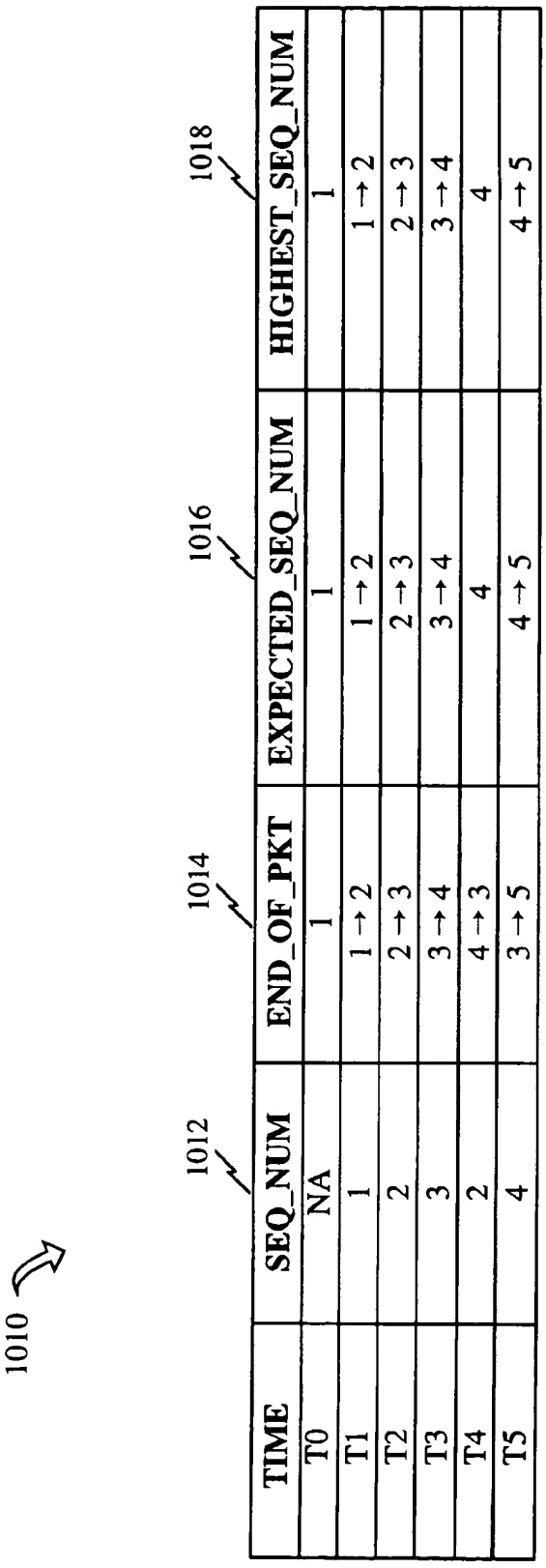

1010
1012
1014
1016
1018
TIME
SEQ_NUM
END_OF_PKT
EXPECTED_SEQ_NUM
HIGHEST_SEQ_NUM
T0
NA
1
1
1
T1
1
1 → 2
1 → 2
1 → 2
T2
2
2 → 3
2 → 3
2 → 3
T3
3
3 → 4
3 → 4
3 → 4
T4
2
4 → 3
4
4
T5
4
3 → 5
4 → 5
4 → 5

FIG. 10

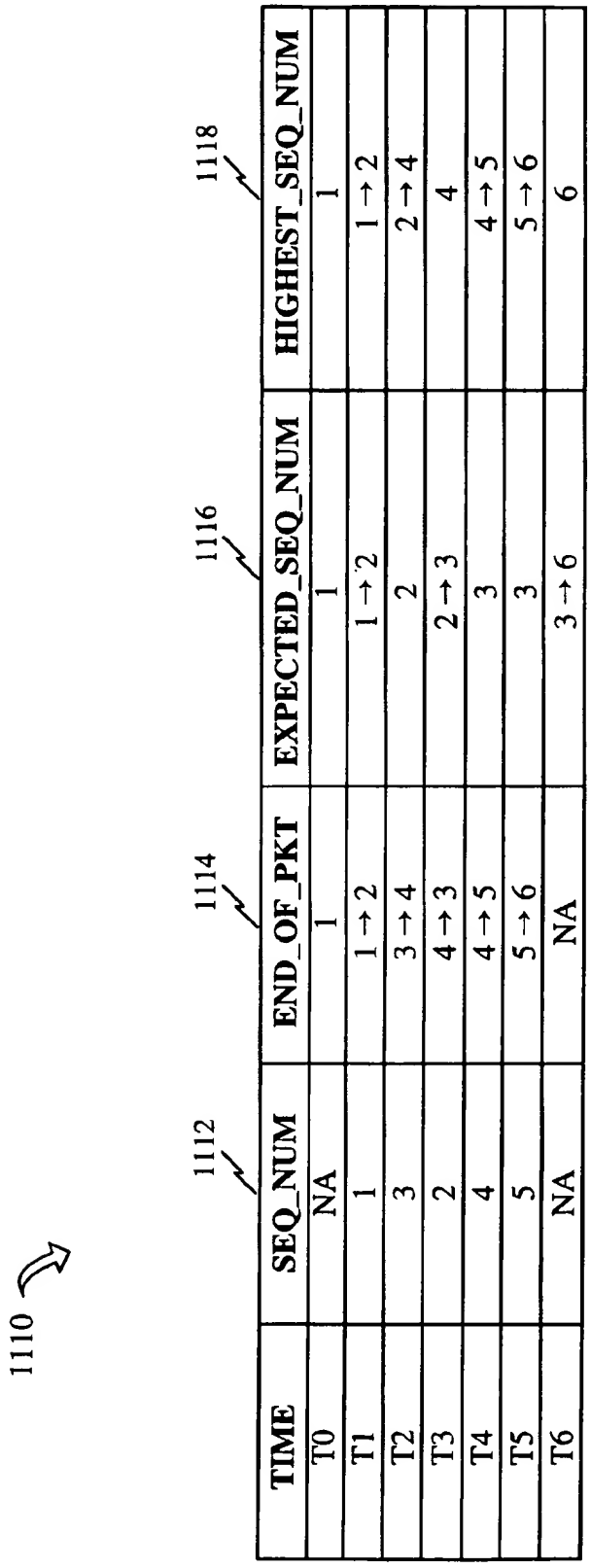

1110
1112
1114
1116
1118
TIME
SEQ_NUM
END_OF_PKT
EXPECTED_SEQ_NUM
HIGHEST_SEQ_NUM
T0
NA
1
1
1
T1
1
1 → 2
1 → 2
1 → 2
T2
3
3 → 4
2
2 → 4
T3
2
4 → 3
2 → 3
4
T4
4
4 → 5
3
4 → 5
T5
5
5 → 6
3
5 → 6
T6
NA
NA
3 → 6
6

FIG. 11

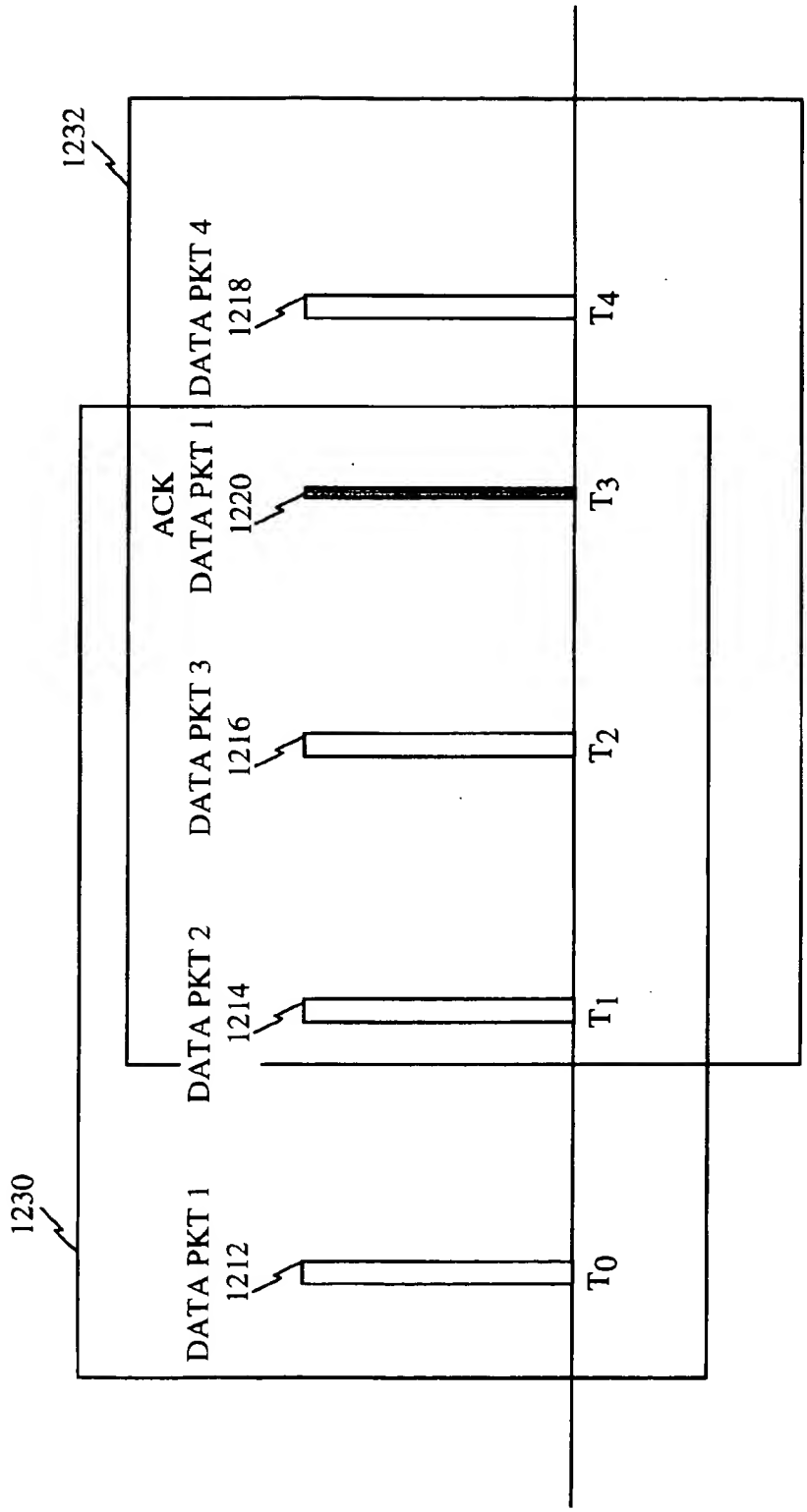

1232
DATA PKT 4
1218
T4
ACK
DATA PKT 1
1220
T3
DATA PKT 3
1216
T2
DATA PKT 2
1214
T1
1230
DATA PKT 1
1212
T0

FIG. 12

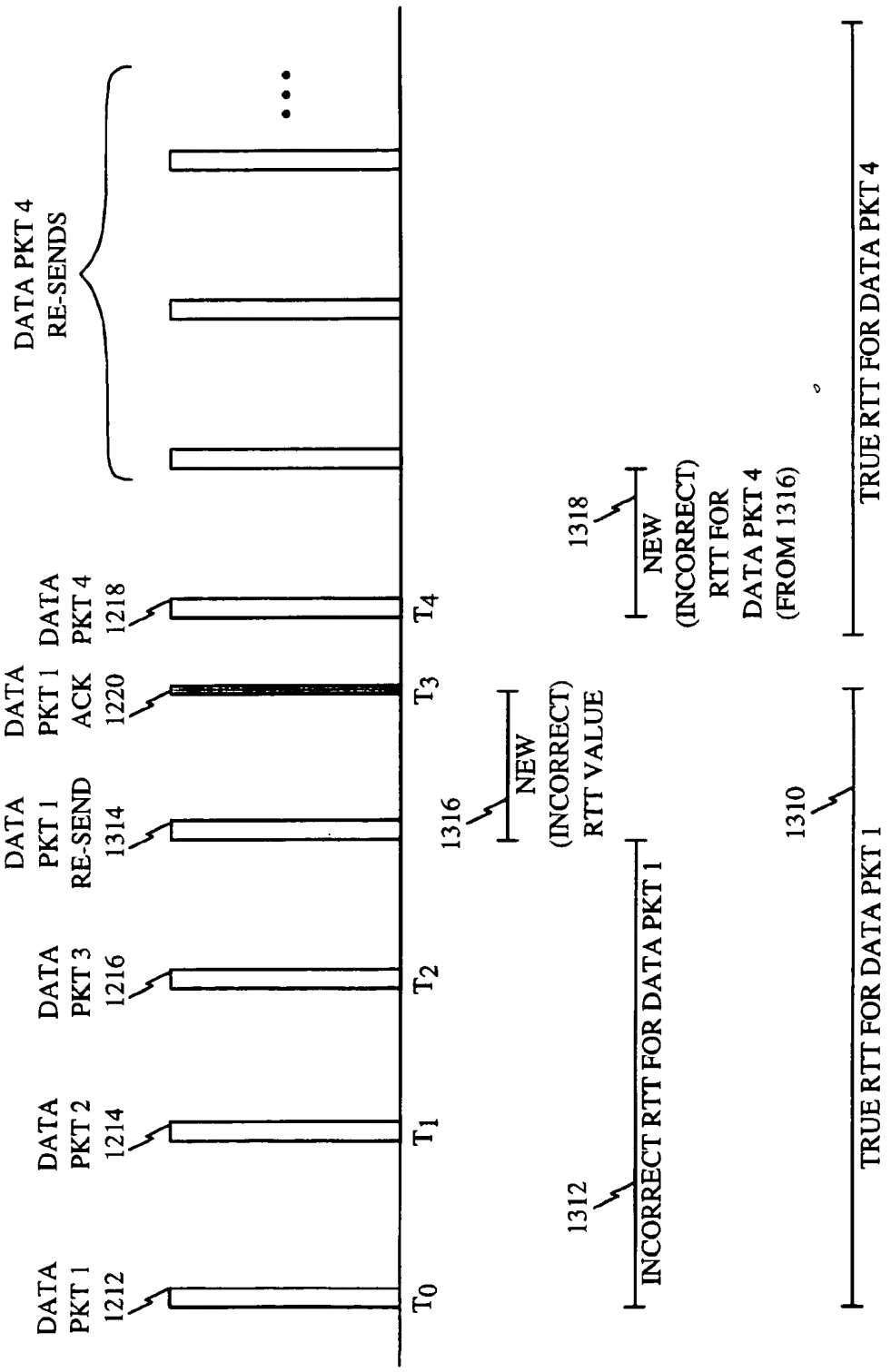

DATA PKT 1
1212
DATA PKT 2
1214
DATA PKT 3
1216
DATA PKT 1 RE-SEND
1314
DATA PKT 1 ACK
1220
DATA PKT 4
1218
DATA PKT 4 RE-SENDS
T0
T1
T2
T3
T4
1312
INCORRECT RTT FOR DATA PKT 1
1316
NEW (INCORRECT) RTT VALUE
1318
NEW (INCORRECT) RTT FOR DATA PKT 4 (FROM 1316)
1310
TRUE RTT FOR DATA PKT 1
TRUE RTT FOR DATA PKT 4

FIG. 13

SYSTEM METHOD AND COMPUTER PROGRAM PRODUCT FOR ELIMINATING UNNECESSARY RETRANSMISSIONS

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention is related to elimination of unnecessary data retransmissions. More particularly, the present invention is related to preventing unnecessary retransmissions from being sent from a terminal, such as an internet host computer, on a slow link.

2. Related Art

Data communications systems typically employ a send and acknowledge scheme, whereby a first terminal sends data packets to a second terminal and the second terminal responds with acknowledge signals. Generally, if the first terminal does not receive an acknowledge signal within a prescribed period of time, the first terminal retransmits the data packet under the assumption that the initial transmission was not received by the second terminal.

The prescribed period of time is often based on an expected round trip time it takes for the data packet to travel to the second terminal and for the corresponding acknowledge signal to return to the first terminal. Round trip times are typically estimated by sending a sample data packet and measuring the time it takes to receive an acknowledge.

Some communication links include what are referred to herein as "slow links." Slow links are communication links that add some amount of latency to the round trip time, wherein the latency depends upon the amount of data being sent. Small sample data packets are often unaffected by latency of slow links. Estimates of round trip times that rely only on small data packet round trip times thus underestimate actual round trip times. As a result, round trip times expire prematurely, causing unnecessary retransmissions of data. Unnecessary retransmissions clog communication links and cause other problems as well. In many situations, the user of the second terminal has no control over the first terminal and thus has no ability to require or ensure that the first terminal correctly estimates round trip time.

Unnecessary data retransmissions are a problem for many types of data communications. In order to more fully understand the cause, effects and solution to unnecessary data retransmissions, data retransmissions are described below within the context of an internet communication system. The examples below are provided to illustrate the problem and solution of unnecessary data retransmissions. They are not intended to limit the scope of the present invention.

An internet communication system includes interconnected networks that support communication among host computers using internet protocols. A host computer typically executes application programs on behalf of users, employing network and/or internet communication services in support of this function. Internet host computers can span a wide range of sizes, speed and functions and can range in size from small microprocessors through workstations to mainframes and super computers. In function, they range from single-purpose hosts (such as terminal servers) to full-service hosts that support a variety of online network services, typically including remote login, file transfer and electronic mail.

The internet is thus a network of networks where each host is directly connected to some particular network or networks. A connection between a host and the internet is only conceptual. Two hosts on the same network communicate with each other using the same setup protocols that they would use to communicate with hosts on distant networks.

Individual or remote users can access the internet from remote terminals (e.g., personal computers, laptop computers, etc.) via internet service providers (ISP). An ISP can also serve as an internet host computer. Remote users are typically connected to ISPs through a public service telecommunications network (PSTN). A coupling between a remote user and a PSTN is typically a slow link such as a PSTN cellular link or a code division multiple access (CDMA) cellular link, which can limit transmission to nominal rates such as, for example, 1200 bits per second (bps), 2400 bps, 4800 bps, 9600 bps, 14,400 bps, etc. These systems are referred to as slow links because communications through these slow links tend to be slower than communications between hosts and between hosts and ISPs on the internet.

When a host computer sends a stream of data packets through an ISP to a remote terminal, the slow link between the ISP and the remote terminal creates a bottleneck where data packets from the host congregate while waiting to be sent to the remote terminal over the slow link. As a result, slow links limit data throughput.

A basic objective of internet design is to tolerate a wide range of network characteristics such as, for example, bandwidth, delay, packet loss, packet reordering and maximum packet size. Accordingly, an internet engineering task force has generated communication specifications for internet host computers. Some of these specifications are available as Request For Comments (RFCs), that are available on the internet at, for example, http://www.nexor.com/ public/rfc/rfcs/rfc. Of particular interest here is an RFC 1122, available on the internet at http://www.nexor.com/public/rfc/rfcs/rfc1122.txt, incorporated herein in its entirety by reference.

These RFCs provide specifications for, among other things, a transmission control protocol (TCP) used by internet host computers. TCP is the primary protocol for the internet. TCP provides reliable, in-sequence delivery of a full-duplex stream of octets (8-bit bytes). TCP is used by applications that require reliable, connection-oriented transport service, e.g., mail (SMTP), file transfer (FTP), and virtual terminal service (Telnet).

Host computers typically maintain a sliding window within which they transmit data packets. Under RFC 1122, in order to prevent bottlenecks, host computers wait for an acknowledgment from a remote terminal when a data packet is sent. Host computers use the acknowledge signals to slide the window and transmit more data.

Host computers typically set a round trip timer based on an estimated round trip time that it takes for a data packet to travel to a remote user and for an acknowledgment from the remote user to travel back to the host computer. When a host computer sends a data packet, the round trip timer is set. If an acknowledgment is not received by the expiration of the round trip timer, the data packet is resent on the assumption that the data packet was not received by the remote terminal. According to RFC 1122, section 4.2.2.15, titled "Retransmission Timeout: RFC-793 section 3.7",

> Recent Work by Jacobson [TCP:7] on Internet Congestion and TCP Retransmission Stability has produced a transmission algorithm combining "slow start" with "congestion avoidance". A TCP MUST implement this algorithm.

For purposes of the present invention, it is assumed that, when a host computer implements a retransmission timeout

algorithm as required by RFC 1122, transmissions from internet host computers to remote users are optimized. In other words, it is assumed that where a host computer follows the requirements of RFC 1122, round trip times are properly estimated.

However, it has been determined that at least some internet host computers (hereinafter "non-compliant host computers" or "non-compliant hosts") do not employ, or do not properly employ, algorithms for optimizing data retransmissions. More specifically, it is believed that non-compliant host computers do not take slow links into account when setting round trip timers.

A host computer should calculate round trip time based on the sum of the actual round trip time of a small (e.g., 1 byte) data packet and based on any slow link latency in the system. Small data packets are typically not affected by slow links because their small size permits them to be sent over the slow link in a single send operation and because they are typically the first data sent to a slow link. Being the first data sent to a slow link means there is no existing bottle neck at the slow link which might otherwise slow transmission of the test data.

Based on examination of actual transmitted data packets, it appears that non-compliant host computers set their round trip timers based only on the round trip time of a small (e.g., 1 byte) test packet. When a non-compliant host computer thereafter sends non-test data packets, which are typically much larger than 1 byte, the data packets get delayed at the slow link. Since the non-compliant host failed to take the slow link into account, the round trip timer expires before receipt of an acknowledgment signal from the remote terminal. The non-compliant host then retransmits the data packet under the assumption that the first data packet was not received by the remote terminal. Unnecessary retransmitted data packets lead to additional latency at the slow link.

In addition to increasing the delay at slow links, non-compliant hosts also mistakenly re-estimate round trip times based on acknowledgment data. When a non-compliant host sends a data packet, it also incorrectly sets its round trip timer. As a result of incorrectly setting its round trip timer, the round trip timer expires before receipt of an acknowledge from the remote terminal. When the round trip timer expires, the non-compliant host retransmits the data packet and resets the round trip timer.

When the first data packet is eventually received by the remote terminal, the remote terminal sends an acknowledge signal. Since the non-compliant host computer receives the acknowledge after retransmitting the data packet and after resetting the round trip timer, the non-compliant host mistakenly interprets the acknowledge as an acknowledgment of receipt of the retransmitted data packet rather than the initial data packet. The non-compliant host then re-estimates the round trip time as the time when it retransmitted the data packet to the time it received the acknowledge to the initial data packet. As a result, the non-compliant host expects to receive acknowledgment of subsequently transmitted data packets faster than before and, thus, retransmits subsequent data packets at a higher rate than before, further adding to the bottleneck at the slow link. It has been observed that unnecessary retransmissions from a non-compliant host computer can reduce the effective rate of data transmission from a nominal rate of 14.4 kbps to an actual data rate of 4–5 kbps or less.

One solution is to ensure that all internet host computers implement a compliant round trip timer scheme such as, for example, that specified in RFC 1122. However, no such mechanism or procedure currently exists.

What is needed is a system, method and computer program product for optimizing data transfers from host computers over slow links.

SUMMARY OF THE INVENTION

The present invention is a novel and improved system, method and computer program product for eliminating unnecessary retransmissions of data packets that are sent from, for example, an internet host computer. The present invention receives data packets from a host computer and determines, based on an estimated round trip time, whether a data packet is a new data packet, a valid retransmitted data packet or an unnecessary retransmitted data packet. New data packets and valid retransmitted data packets (i.e., out-of-order data packets), are forwarded to a remote terminal. Unnecessary retransmitted data packets are discarded.

The estimate of round trip time is determined as the time it takes for a data packet to travel to the remote terminal and for an acknowledgment of receipt to return from the remote terminal. The estimated round trip time includes throughput latency that results from a slow link.

In one embodiment, the present invention is implemented as a slow link TCP optimizer that includes a round trip timer, data tables that store information associated with the data packets including an expected sequence number, and a discriminator that employs the round trip timer to determine whether the data packets are valid data packets or unnecessary retransmitted data packets.

In one embodiment, the slow link TCP optimizer treats a data packet as a valid data packet if the round trip timer has expired. In addition, the slow link TCP optimizer determines that a data packet is a valid data packet when the round trip timer has not expired and the data packet has a sequence number that is not less than the expected sequence number. The slow link TCP optimizer also determines that a data packet is a valid data packet when the round trip timer has not expired, the data packet has a sequence number that is less than the expected sequence number and the data packet includes data that is not substantially similar to data in a previously received data packet. The slow link TCP optimizer determines that a data packet is an unnecessary retransmitted data packet when the round trip timer has not expired, the data packet has a sequence number that is less than the expected sequence number and the data packet includes data that is substantially similar to data in a previously received data packet.

In one embodiment, the data tables include a current sequence number, an end of packet number, a highest sequence number and an expected sequence number. When a data packet is a valid data packet, the slow link TCP optimizer updates the data tables as follows: the end of packet number is set to the sum of the TCP sequence number of the data packet and the IP length field of the data packet; the expected sequence number is set to the end of packet number when the sequence number is equal to the expected sequence number; the highest sequence number is set to the end of packet number when the end of packet number is greater than the highest sequence number; and the expected sequence number is set to the highest sequence number when the data packet has a sequence number that is less than the expected sequence number and the data packet includes data that is not substantially similar to data in a previously received data packet. Additionally, when a data packet is an unnecessary retransmitted data packet, the slow link TCP optimizer updates the data tables by setting the expected

sequence number to the highest sequence number. Also, when a timer expires, the expected sequence number is set to be equal to the highest sequence number.

The present invention can be implemented in software, firmware, hardware or any combination thereof. The present invention is implemented between a slow link and a host computer. For example, the present invention can be implemented between a slow link and an internet service provider (ISP), within an ISP, between an ISP and a host, etc. The present invention can be employed by, for example, a telecommunications facility that provides communications between slow links (i.e., modems) and ISPs and by cellular telecommunications facilities that provide slow links between mobile telecommunication units and public service telephone networks (PSTNs).

One advantage of the present invention is that it eliminates unnecessary retransmissions that are sent from a host computer. Another advantage of the present invention is that it accommodates out-of-order data packets. Another advantage of the present invention is that it does not affect data packets that are sent from a host computer that properly accounts for a slow link when determining a round trip time.

Further features, objects, and advantages of the present invention will become more apparent from the detailed description set forth below.

BRIEF DESCRIPTION OF THE DRAWINGS

The features, objects, and advantages of the present invention will become more apparent from the detailed description set forth below when taken in conjunction with the drawings in which like reference characters identify correspondingly throughout. Additionally, the left-most digit of a reference number identifies the drawing in which the reference number first appears.

FIG. **1** is a block diagram of an interconnection between an internet host computer and a remote terminal through a slow link;

FIG. **2** is a code division multiple access (CDMA) cellular telecommunications example of the slow link illustrated in FIG. **1**;

FIG. **3** is a block diagram of a CDPD cellular telecommunications example of the slow link illustrated in FIG. **1**;

FIG. **4** is a land line modem/PSTN example of the slow link illustrated in FIG. **1**;

FIG. **5** is a block diagram of a slow link TCP optimizer that eliminates unnecessary retransmissions from a host computer, in accordance with the present invention;

FIG. **6** is a block diagram of the slow link TCP optimizer of FIG. **5**, implemented in a CDMA system;

FIG. **7** is a block diagram of the slow link TCP optimizer of FIG. **5**, implemented in a land line modem/PSTN system;

FIG. **8** is a high-level process flowchart illustrating a method for eliminating unnecessary retransmissions from a host computer, in accordance with the present invention;

FIG. **9** is a detailed process flowchart illustrating a method for eliminating unnecessary retransmissions from a host computer, in accordance with the present invention;

FIG. **10** is a data table that can be employed by the slow link TCP optimizer illustrated in FIG. **5**;

FIG. **11** is a data table that can be employed by the slow link optimizer illustrated in FIG. **5**;

FIG. **12** is a time-line of data packets transmitted from a host computer; and

FIG. **13** is a time-line of data packets transmitted from a non- compliant host computer.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

1. Overview of the Invention

The present invention is a system, method and computer program product for preventing unnecessary retransmissions from being sent from a terminal such as, for example, an internet host computer, on a slow link.

The present invention receives data packets from a host computer and determines, based on an estimated round trip time, whether a data packet is a new data packet, a valid retransmitted data packet or an unnecessary retransmitted data packet. New data packets and valid retransmitted data packets are sent to the remote terminal. Unnecessary retransmitted data packets are discarded. The present invention also accommodates out-of-order data packets.

The present invention does not affect data packets that are sent from a compliant host computer that properly accounts for slow links. Thus, the present invention can be employed between a slow link and the internet without regard to whether a host computer properly accounts for a slow link. For example, the present invention can be employed by a telecommunications facility that provides communications between slow links (i.e., modems) and ISPs. The present invention can also be employed by a cellular telecommunications facility, such as a code division multiple access (CDMA) system, that provides a slow link connection between mobile telecommunication units and the Internet.

2. Example Environment

The present invention can be employed in any data communication system that retransmits data when an acknowledge signal is not received within a period of time. In one embodiment, the present invention is employed to prevent sending of unnecessary retransmissions from an internet host computer on a slow link. A system, method and computer program product for preventing unnecessary retransmissions from being sent from an internet host computer on a slow link, are now described. The examples herein are provided to assist in the description of the present invention, not to limit it.

To communicate using the internet system, a host computer can implement a layered set of protocols that is referred to in RFC 1122 as the internet protocol suite. A host computer typically implements at least one protocol from each layer. The protocol layers that are employed by the internet include an application layer, a transport layer, an internet layer and a link layer.

The application layer is the top layer of the internet protocol suite. The application layer includes user protocols that provide service directly to users and support protocols that provide common system functions. Typical internet user protocols include Telnet for remote login, File Transfer Protocol (FTP) for file transfers and SMTP for electronic mail delivery. There can also be a number of other standardized user protocols and private user protocols.

The transport layer provides end-to-end communication services for applications. Two primary transport layer protocols include Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP is a reliable connection-oriented transport service that provides end-to-end reliability, re-sequencing and flow control. UDP is a connection list ("datagram") transport service. Other transport protocols have been developed by internet research communities. Internet transport protocols can vary.

In the internet layer, all internet transport protocols use an internet protocol (IP) to carry data from source host to

destination host. An IP datagram is the unit of end-to-end transmission in the IP protocol. An IP datagram includes an IP header followed by transport layer data, i.e., an IP header followed by a message. IP is a connection list or datagram internet work service, providing no end-to-end delivery guarantees. Thus, IP datagrams can arrive at the destination host damaged, duplicated, out-of-order or not at all. The layers above IP are responsible for reliable delivery service when it is required. The IP protocol includes provisions for addressing, type of service specification, fragmentation and reassembly, and security information. The internet layer can also employ an internet control message protocol (ICMP) and an IGN protocol (IGNP).

In the link layer, in order to communicate on its directly-connected network, a host computer can implement the communication protocol that is used to interface to that network. The link layer can be referred to as a media access layer. There is a wide variety of link layer, or media access layer, protocols, corresponding to the many different types of networks.

Referring to FIG. **1**, a remote terminal **110** is connected to the internet **112** through a slow link **116** and an internet service provider (ISP) **114**. Remote terminal **110** accesses an end of network host computer such as host computer **118** that is coupled to internet **112**. End of network host computer **118** sends data packets **120** to remote terminal **110** through internet **112**, ISP **114** and slow link **116**. When remote terminal **110** receives a data packet **120**, it sends an acknowledge **122** back to host computer **118**. Within internet **112**, the path taken by acknowledge signals **122** can be different than the path taken by data packets **120**.

Referring to FIG. **2**, slow link **116** is illustrated as a CDMA cellular telephone interconnection wherein ISP **114** sends data packets to a standard network router **212** which forwards the data packets to a CDMA inner working functions (IWF) module **214**. IWP **214** acts as a router between a CDMA network and the internet. IWP **214** sees and understands every TCP IP header that comes through it. CDMA IWF **214** sends the data packets to a base station controller **218** for RF transmission to a CDMA mobile unit **216** which is coupled to remote terminal **110**.

Referring to FIG. **3**, slow link **116** is illustrated as a cellular digital packet data (CDPD) system including a CDPD mobile unit **310** and a CDPD mobile data intermediate system (MDIS) unit **312**.

Referring to FIG. **4**, slow link **116** is illustrated as a land line modem/PSTN interconnection between remote terminal **110** and a standard network router **212**. The land line modem/PSTN interconnection includes a land line modem **412** coupled to a PSTN **410** which is coupled to a standard network router **212** through a server **414**.

In order to prevent excessive queuing of data packets **120** as shown in FIG. **1** at slow link **116**, host computer **118** typically implements an acknowledge scheme that sends up to a given number of data packets, or a given amount of data, within a sliding window. When an acknowledgment of receipt of a data packet is received, the window slides so that another data packet can be sent.

For example, referring to the time-line of FIG. **12**, using a sliding window **1230**, host computer **118** can have, for example, up to three data packets outstanding at a time. Thus, host computer **118** can send data packet **1212** at time T0, followed by data packet **1214** at time T1 and data packet **1216** at time T2. Host computer **118** then waits for acknowledges **1220** from remote terminal **110**. When host computer receives an acknowledge for any of data packets **1212–1216**, window **1230** slides to permit another data packet to be sent by host computer **118**. For example, when an acknowledge **1220** is received at time T3 for data packet **1212**, sliding window **1230** becomes sliding window **1232** and host computer **118** can send a fourth data packet **1218** at time T4.

Host computer **118** sets a round trip timer (RTT) (not shown) for each data packet that it sends. The RTT is an estimated round trip time that it takes for a data packet **120** to be received by remote terminal **110** and for a corresponding acknowledge **122** to be received by host computer **118**. If the RTT for a data packet **120** expires before receipt of acknowledge **122**, host computer **118** presumes that the data packet **120** was lost prior to receipt by remote terminal **110**. Host computer **118** then retransmits the data packet to remote terminal **110**.

Host computer **118** should calculate the RTT based on the sum of individual delays of the transmit chain plus any latency due to slow links, such as slow link **116**. Host computer **118** can, for example, send a 1 byte packet to remote terminal **110** and measure the time between sending the packet and receiving an acknowledge. In FIG. **2**, for example, there can be 80 milli-seconds (msec) of delay in CDMA mobile unit **216**, 20 msec of delay between CDMA mobile **216** and base station controller **218**, 40 msec delay in base station controller **218** and CDMA IWF **214** and 80 msec delay between host **118** and CDMA IWF **214**, for a total one way trip delay of 220 msec or a round trip delay of 440 msec.

Slow link **116** also has a latency or bottle neck throughput which can be, for example, 1200 bits per second (bps), 2400 bps, 4800 bps, 9600 bps, 14,400 bps, etc. When a data packet is relatively small (e.g., 1 byte), the test data packet is affected by the slow link latency, but only by the delay described above. However, larger data packets and data packets that get queued in the slow link behind other data packets will be subjected to slow link latency that can vary with the slow link nominal throughput, the size of the data packet and size of the slow link queue.

In order to properly determine RTT, host computer **118** must estimate the slow link latency as well as the normal delay. Failure to do so will result in under estimating RTT. When RTT is under estimated, RTT will time out before a corresponding acknowledge **122** is received by host computer **118**.

For example, referring to the time-line of FIG. **13**, RTT **1310** is the actual round trip time for data packet **1212**. RTT **1312** is an incorrectly estimated round trip time that omits the latency due to slow link **116**. RTT **1310** is not necessarily drawn to scale with RTT **1312**. At the end of RTT **1312**, host computer **118** retransmits data packet **1212** as data packet **1314** because it did not receive an acknowledge for data packet **1212** before RTT **1312** expired. Unless data packet **1212** was actually lost in transmission, retransmitted data packet **1314** is unnecessary and thus, adds to the bottleneck at slow link **116**.

An additional problem caused by a non-compliant host computer **118** is a further shortening of RTT **1312** based on acknowledge **1220**. Host computer **118** receives acknowledge **1220** at time T3 as an acknowledgment of receipt of data packet **1212**. However, since data packet **1314** is a retransmission of data packet **1212**, acknowledge **1220** of data packet **1212** is substantially identical to an acknowledge that would be expected by host computer **118** for data packet **1314**. Since host computer **118** reset its RTT for the data packet when it retransmitted it as **1314**, host computer **118** concludes that acknowledge **1220** is an acknowledge of

retransmitted data packet **1314**. Host computer **118** then re-estimates RTT based on the elapsed time **1316** between retransmission **1314** and acknowledge **1220**. As a result, when host computer **118** sends data packet **1218** at time T4, RTT is set to new RTT **1318** (eg., a value equivalent to **1316**), which is even shorter than the originally miscalculated RTT **1312**. Thus, an acknowledge for data packet **1218** will be expected within new RTT **1318**. When host computer **118** fails to received an acknowledge within new RTT **1318**, it will re-send data packet **1218** until it receives an acknowledge.

In a slightly more complicated scenario, which has been experienced, incorrect RTT **1312** and/or new RTT **1318** are much shorter than actual RTT **1310**. As a result, host computer **118** unnecessarily retransmits a given data packet multiple times before receiving an acknowledge for the initial sending of the data. When this scenario is repeated for data packets **1214** and **1216** as well, the bottleneck of slow link **116** can substantially reduce the amount of new data sent from host computer **118** to remote terminal **110**.

3. Slow Link TCP Optimizer

Referring to FIG. **5**, a slow link TCP optimizer **510** eliminates unnecessary retransmissions from a host computer based on a round trip timer (RTT) **512**, which is completely independent of a round trip timer in host computer. Slow link TCP optimizer **510** can be implemented through software, firmware, hardware or any combination thereof. Slow link TCP optimizer **510** includes a discriminator **514** that receives data packets **120** from internet host computer **118** and data tables **516** that store limited information for use by discriminator **514** to determine whether a packet **120** is a new packet, a valid retransmitted packet or an unnecessary retransmitted packet.

In one embodiment, data tables **516** store a current sequence number (seq_num), an end-of-packet number (end_of_pkt), an expected sequence number (expected_seq_num) and a highest sequence number (highest_seq_num). These values are generated from a TCP sequence number and an IP field length that are included in each data packet.

Referring to FIG. **10**, sample values for a current seq_number **1012**, an end_of_pkt number **1014**, an expected_seq_num **1016** and a highest_seq_num **1018** are illustrated for consecutive data packets **120** that are received by slow link TCP optimizer **510** at times T0 through T5. In this embodiment, discriminator **514** compares the sequence number of a received packet **120** to one or more of data values **1014–1018**, to determine whether the received data packet is a new packet, a valid retransmitted packet or an unnecessary retransmitted packet.

Data tables **516** typically store these values only for a current time. The consecutive values in FIG. **10** are provided to illustrate the changing values over time, as described more fully below with reference to the process flowchart of FIG. **9**.

Slow link TCP optimizer **510** does not store a queue of data packets **120**. Instead, discriminator **514** determines whether a data packet is valid based on round trip timer **512** and data tables **516**. This eliminates a need for a buffer queue which could otherwise be employed for determining whether a data packet **120** is a valid packet, a valid retransmitted packet or an unnecessary retransmitted packet.

Slow link TCP optimizer **510** passes valid packets and valid retransmitted packets to slow link **116**, as data packets **518**, for transmission and to remote terminal **110**. Slow link TCP optimizer **510** discards unnecessary retransmitted data packets. Unnecessary transmissions thus do not contribute to bottleneck at slow link **116**. Additional details of slow link TCP optimizer **510** are provided below with reference to the process flowcharts of FIGS. **8** and **9**.

Referring to FIG. **6**, slow link TCP optimizer **510** can be implemented in a CDMA system **610**, which can be the CDMA system illustrated in FIG. **2**. In FIG. **6**, slow link TCP optimizer **510** is illustrated as implemented within IWF **214**. Alternatively, slow link TCP optimizer **510** could be implemented between IWF **214** and base station controller **218**, between IWF **214** and ISP **114**, between ISP **114** and internet **112** or within ISP **114**.

Referring to FIG. **7**, slow link TCP optimizer **510** can be implemented in a land line modem/PSTN system **710** such as that illustrated in FIG. **4**. In FIG. **7**, slow link TCP optimizer **510** is illustrated as implemented within ISP **114**. Alternatively, slow link TCP optimizer **510** could be implemented as a stand-alone system between PSTN **410** and ISP **114**, between ISP **114** and internet **112** or within PSTN **410**.

Referring to FIG. **8**, a process flowchart is illustrated for eliminating unnecessary retransmissions from a host computer. The process flowchart of FIG. **8** is described as implemented by TCP optimizer **510**. As will be apparent to one skilled in the art, the process flowchart of FIG. **8** could be implemented by a variety of systems and computer program products. The use of slow link TCP optimizer **510** for the description of the process flowchart of FIG. **8** is intended as an example to illustrate the present invention, not to limit it.

The steps illustrated in FIGS. **8** and **9** do not necessarily occur in the same order as illustrated in pseudo code **1410**. The exact order with which the steps are performed can be rearranged, as will be recognized by one skilled in the art, so long as a determination as to whether to forward a data packet **120** to remote terminal **110** is based on round trip timer **512**.

In FIG. **8**, the process begins at step **810**, where round trip timer **512** is initialized to zero, for example. A separate RTT timer **512** is initialized for every session and for each remote terminal **110**. For example, when a CDMA mobile unit **216** initiates a session with base station controller **218**, a round trip timer **512** is set for that session.

In step **812**, discriminator **514** receives a data packet **120** from host computer **118** and records data in data tables **516**. The data packet can be, for example, one of data packets **1212**, **1214**, **1216**, **1218** or **1314**. The recorded data preferably includes data packet sequence number and header information. This data is used in step **824**, as described more fully below.

In step **814**, discriminator **514** determines whether RTT **512** has expired. If RTT **512** has expired, processing proceeds to step **815**, where expected_seq_num **1016** is set to highest_seq_num.

In step **816**, the data packet is sent as a valid data packet **518** to remote terminal **110**. Thus, if RTT **512** has expired, the data packet is automatically treated as a valid data packet, regardless of whether it is a first transmission of the data packet or a retransmission of a prior data packet. More specifically, since RTT **512** has expired, it is concluded that all data packets that were previously sent to slow link **116** have either been received and acknowledged by remote terminal **110** or lost in transmission. Since compliant host computers **118**, by definition, do not unnecessarily retransmit data packets before RTT **512** expires, the present invention has no effect on data packets that are sent from compliant host computers.

In step **818**, RTT **512** is reset to an estimated round trip time that it takes a data packet **518** to travel from slow link TCP optimizer **510** to remote terminal **110** and for an acknowledge **122** to travel from remote terminal **110** back to slow link TCP optimizer **510**. Round trip timer **512** can be set according to RFC 1122 or any other suitable algorithm, so long as it sums nominal transmission delay with any slow link latency.

In step **820**, if another data packet is received, control passes back to step **812** for receipt of the data packet. Otherwise, processing stops at step **822**.

Referring back to step **814**, if RTT **512** has not expired, processing proceeds to step **824** where discriminator **514** determines whether the received packet is a new packet or a valid retransmitted packet. Discriminator **514** can, for example, compare a sequence number of the received packet to data in data tables **516**.

Referring to the process flowchart of FIG. **9** and data table **1010** (see FIG. **10**), a preferred method for implementing step **824** is provided. Processing begins at step **910**, where discriminator **514** compares the sequence number of the received data packet, i.e., seq_num **1012**, to expected_seq_num **1016**. Preferably, if the received data packet is the first data packet in a sequence, slow link TCP optimizer **510** initializes end-of-packet value **1014**, expected sequence number value **1016** and highest sequence number **1018**, as illustrated at time T0.

For example, in FIG. **10**, a data packet having a sequence number **1** is received and recorded at time T1 in step **812**. In step **910** discriminator **514** determines whether seq_num **1012** is less than expected sequence number **1016**. Since seq_num **1012** for T1 is not less than expected_seq_num **1016** for T1, processing proceeds to step **912**, where end_of_pkt **1014** is set to the sum of the current packet TCP seq_num **1012** plus the current packet's IP field length. In other words, end-of-packet value **1014** is set to the end of the present sequence number which is, essentially, the beginning of the next logical sequence number. Thus, at time T1, end_of_pkt **1014** is set to 2.

In step **914**, discriminator **514** determines whether current seq_num **1012** is equal to expected_seq_num **1016**. Where, as at time T1, current seq_num **1012** T1 is equal to expected_seq_num **1016**, processing proceeds to step **916** where expected_seq_num **1016** is set to end_of_pkt **1014**. Thus, at time T1, expected_seq_num **1016** is shown changing from 1 to 2.

In step **918**, discriminator **514** determines whether end_of_pkt **1014** is greater than highest_seq_num **1018**. Where, as at time T1, end_of_pkt **1014** is greater than highest_seq_num **1018**, processing proceeds to step **920** where highest_seq_num **1018** is set to end_of_pkt **1014**.

In step **922**, discriminator **514** determines whether the received packet was declared a valid retransmission. Declaration of a packet as a valid retransmission is discussed more fully below in step **936**. Where, as at time T1, the received data packet is a new transmission, processing proceeds to step **816** for sending of the packet.

When discriminator **514** receives another packet in step **812**, and if RTT **512** has not expired, processing returns to step **910** where discriminator **514** determines whether the sequence number of the newly received packet is less than the expected sequence number **1016**.

For example, referring to time T2 in FIG. **10**, if the sequence number of the newly received data packet is sequence number **2**, processing proceeds just as with time T1. As long as each subsequently received packet is the next expected packet, data table **1010** is updated as at time T1 and discriminator **514** passes data packets **120** to remote terminal **110** as data packets **518**. Thus, so long as host computer **118** properly sets its internal round trip timer, it does not send any unnecessary retransmissions and discriminator **514** does nothing more than update data tables **516** and pass data packets **120** to remote terminal **110** as data packets **518**.

However, if internet host **118** retransmits a data packet, discriminator **514** employs round trip timer **512** and data tables **516** to determine whether the retransmission is a valid retransmission or an unnecessary retransmission. For example, referring to time T3 in FIG. **10**, the next expected data packet has a sequence number of **3**. Accordingly, end_of_pkt **1014**, expected_seq_num **1016** and highest seq_num **1018** are updated as at time T1. Thus, at the end of time T3, end-of-packet value **1014**, expected sequence number **1016** and highest sequence number **1018** are all set to 4 to indicate that the next expected sequence number is **4**.

However, at time T4, the next packet received in step **812** is sequence number **2**. Thus, in step **910**, discriminator **514** determines that seq_num **1012** is less than expected_seq_num **1016** and processing proceeds to step **928** where discriminator **514** declares the packet a retransmission. This does not automatically preclude the data packet from being sent to remote terminal **110**. Instead, processing proceeds to step **930** where discriminator **514** determines whether there are any significant changes between the current packet and the prior packet. Discriminator **514** can compare, for example, header information of the current packet and header information of the prior data packet or prior data packets.

If there are no significant changes, processing proceeds to step **932**, where discriminator **514** declares that the packet is an unnecessary retransmission. In step **934**, expected_seq_num **1016** is set to highest_seq_num **1018**. The significance of step **934** is discussed below. In step **826**, the data packet is discarded as an unnecessary retransmitted packet and processing proceeds to step **820** for receipt of additional packets.

The present invention is designed to accommodate, and recover from, unnecessary retransmissions, such as the unnecessary retransmission of data packet seq_num **2**, discussed above. Referring to FIG. **10**, at time T5, a data packet having a sequence number of 4 is received in step **812**.

In step **910**, discriminator **514** determines that seq_num **1012** is not less than expected seq_num **1016**. In step **912**, discriminator **514** sets end_of_pkt value **1014** to the sum of the current packet's TCP seq_num plus the current packet's IP length field, in this case **5**.

In step **914**, discriminator **514** determines whether seq_num **1012** is equal to expected_seq_num **1016**. Since seq_num **1012** is equal to expected_seq_num **1016**, processing proceeds to step **916** where discriminator **514** sets expected_seq_num **1016** to end_of_pkt **1014**.

In step **918**, discriminator **514** determines whether end_of_pkt **1014** is greater than highest_seq_num **1018**. Since end_of_pkt **1014** is greater than highest_seq_num **1018**, processing proceeds to step **920** where discriminator **514** sets highest_seq_num **1018** to end_of_pkt **1014**. In step **922**, discriminator **514** determines that the received data packet is not a retransmission and processing proceeds to step **816** for sending of the packet.

The present invention also accommodates out-of-order receipt of data packets. Referring to FIG. **11**, an example of processing of out-of-order data packets is illustrated.

In step **812**, a data packet having a sequence number **1** is received at time T1 and recorded in seq_num **1112**. Accordingly, end_of_pkt **1114**, expected_seq_num **1116** and highest_seq_num **1018** are updated in steps **910–924**, as at time T1 in FIG. **10**.

At time T2, a data packet having a sequence number of **3** is received in step **812** and recorded in seq_num **1112**. In step **910**, discriminator **514** determines that the seq_num **1112** is not less than expected_seq_num **1116** and processing proceeds to step **912**. In step **912**, discriminator **514** sets end_of_pkt **1114** to the sum of the current data packet's TCP sequence number plus the current data packet's IP field length.

In step **914**, discriminator **514** determines that seq_num **1112** is not equal to expected_seq_num **1116**. Processing jumps to step **918** where discriminator **514** determines that end_of_pkt **1114** is greater than highest_seq_num **1118**. In step **920**, discriminator **514** sets highest_seq_num **1118** to end_of_pkt **1114**. In step **922**, discriminator **514** determines that the packet is not a retransmission and processing proceeds to step **816** where the data packet is sent to remote terminal **110**.

At time T3, a data packet having a sequence number of **2** is received in step **812** and recorded in seq_num **1112**. In step **910**, because current seq_num **1112** is two and expected_seq_num **1116** is two, discriminator **514** determines that seq_num **1112** is not less than expected_seq_num **1116**. Processing proceeds to step **912** where end_of_pkt **1114** is set to the sum of the current packet's TCP seq_num and the current packet's IP length field (i.e., three).

In step **914**, discriminator **514** determines that current seq_num **1112** is equal to expected_seq_num **1116** and, in step **916**, sets expected_seq_num **1116** to end_of_pkt number **1114**. In step **918**, discriminator **514** determines that end_of_pkt **1114** is not greater than highest seq_num **118**. Processing jumps to step **922**, where discriminator **514** determines that the packet is not a retransmission. Processing then jumps to step **816** where the packet is sent to remote terminal **110**.

At time T4, and in step **812**, a data packet having a sequence number of four is received and recorded in seq_num **1112**. Provided that RTT **512** has not expired, processing proceeds to step **910** where discriminator **514** determines that seq_num **1112** is not less than expected seq_num **1116**. In step **912**, discriminator **514** sets end_of_pkt **1114** to the sum of the current packet's TCP seq_num **1112** plus the current packet's IP field length. Processing proceeds through step **814** and to step **910**.

In step **910**, discriminator **514** determines that current seq_num **1112** is not greater than expected_seq_num **1116**. In step **912**, end_of_pkt is set to the sum of the current packet's TCP seq_num **1112** plus the current packet's IP field length.

In step **914**, discriminator **514** determines that seq_num **1112** is not equal to expected_seq_num **1116**. Processing jumps to step **918**, where discriminator **514** determines that end_of_pkt **1114** is greater than highest_seq_num **1118**. In step **920**, highest_seq_num **1118** is set to end_of_pkt **1114**.

In step **922**, discriminator **514** determines that the present data packet is not a retransmission. Processing proceeds to step **816**, where the packet is sent to remote terminal **110**.

In step **812**, at time T5, a data packet having a sequence number of five is received and recorded in seq_num **1112**. Provided that RTT **512** has not expired, processing proceeds as at time T4.

Note that from time T3, where packet two was received out of order, expected_seq_number remains at three. When RTT **512** eventually expires, as shown at time T6, for example, processing will reach step **816**, where expected sequence number **1116** is set to highest_seq_num **1118**.

Set forth below is an example of pseudo code that illustrates a method for eliminating unnecessary retransmissions from an internet host computer in accordance with the steps described above with reference to FIGS. **8** and **9**.

```
pkt_hdr=pkt.tcp_ip_hdr;
connection=get_connection(pkt_hdr)//This finds last pkt hdr for this connection
last_pkt_hdr=connection.last_pkt_hdr
rtt=connection.timer
seq_num=pkt.tcp.seq
old_seq_num=last_pkt_hdr.tcp.seq
pkt_is_retx=FALSE;
pkt_valid=TRUE;
if(seq_num<expected_seq_num)
{
    pkt_is_retx=TRUE;
    if(timer_not_expired(rtt))&&(no_significant_hdr_changes(pkt_hdr,
last_pkt_hdr))
        pkt_valid=FALSE
}
if(pkt_valid==FALSE)
{
    expected_seq_num=highest_seq_num;
}
else
{
    set_timer(rtt,(EST_LATENCY+THRUPUT_DELAY
    (pkt_hdr.ip.length)));
    end_of_pkt=seq_num=pkt_hdr.ip.length;   //This is total length of the
                                            IP pkt.
                                            //It is not the header-
                                            length of pkt.
    if(expected_seq_num==seq_num)
    {
```

-continued

```
        expected_seq_num==end_of_pkt;
    }
    if(end_of_pkt>highest_seq_num)
    {
        highest_seq_num=end_of_pkt;
    }
    if(pkt_is_retx)
    {
        expected_seq_num=highest_seq_num;
    }
    connection.last_pkt_hdr=pkt_hdr;
    transmit_the_pkt_in_ppp_format(pkt);
}
```

Not shown in the pseudo code above is a function that is called when RTT timer **512** expires. The function sets expected_seq_num equal to highest_seq_num and sends any packets that are received after RTT timer **512** expires.

4. Conclusions

Based upon the foregoing description, one skilled in the art will recognize that there are many suitable algorithms that could be employed by slow link TCP optimizer **510** for determining whether a data packet is a new data packet, a valid retransmitted data packet or an unnecessary retransmitted data packet without storing a queue of previously received data packets. One skilled in the art will also recognize that there are many suitable algorithms that could be employed by slow link TCP optimizer **510** for discriminating between unnecessary retransmissions and valid apparent retransmissions (i.e., out-of-sequence data packets).

The description of the preferred embodiments is provided to enable any person skilled in the art to make or use the present invention. While the invention has been particularly shown and described with reference to preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention.

What is claimed is:

1. A method for eliminating unnecessary retransmissions of data packets from a first terminal, comprising the steps of:

(1) receiving a data packet from the first terminal at a first location;

(2)(a) estimating at the first location a round trip time for a data packet to travel from the first terminal to a second terminal, and for an acknowledgment of receipt to arrive at the first terminal from the second terminal, wherein the round trip time is based upon a delay in transmission plus throughput latency resulting from a slow link between the first terminal the second terminal;

(2)(b) determining whether the data packet is a valid data packet or an unnecessary retransmitted data packet based upon the estimated round trip time;

(3) forwarding valid data packets to the second terminal; and

(4) discarding unnecessary retransmitted data packets.

2. The method according to claim **1**, wherein step (2)(a) further comprises:

(i) determining that the data packet is a valid data packet if a round trip timer has expired, wherein the round trip timer is set according to the estimated round trip time.

3. The method according to claim **1**, wherein step (2) further comprises:

(a) maintaining a data table including an expected sequence number.

4. The method according to claim **3**, wherein step (2) further comprises the step of:

(b) determining that the data packet is a valid data packet when the following conditions are met;

(i) a round trip timer has not expired, and

(ii) the data packet has a sequence number that is not less than the expected sequence number.

5. The method according to claim **4**, wherein step (2) further comprises the steps of:

(c) maintaining a current sequence number, an end of packet number, a highest sequence number and the expected sequence number in the data table;

(d) setting the end of packet number to a sum of a TCP sequence number of the data packet and an IP length field of the data packet;

(e) setting the expected sequence number to the end of packet number if the sequence number is equal to the expected sequence number; and

(f) setting the highest sequence number to the end of packet number if the end of packet number is greater than the highest sequence number.

6. The method according to claim **3**, wherein step (2) further comprises the step of:

(b) determining that the data packet is a valid data packet when the following conditions are met;

(i) the data packet has a sequence number that is less than the expected sequence number, and

(ii) the data packet includes data that is not substantially similar to data in a previously received data packet.

7. The method according to claim **6**, wherein step (2) further comprises the steps of:

(c) maintaining a current sequence number, an end of packet number, a highest sequence number, the expected sequence number and control flags in the data table;

(d) setting the end of packet number to a sum of a TCP sequence number of the data packet and an IP length field of the data packet;

(e) setting the expected sequence number to the end of packet number if the sequence number is equal to the expected sequence number;

(f) setting the highest sequence number to the end of packet number if the end of packet number is greater than the highest sequence number; and

(g) setting the expected sequence number to the highest sequence number.

8. The method according to claim **3**, wherein step (2) further comprises the step of:

(b) determining that the data packet is an unnecessary retransmitted data packet when;

(i) the data packet has a sequence number that is less than the expected sequence number,

(ii) the data packet includes data that is substantially similar to data in a previously received data packet, and

(iii) the timer is still running.

9. The method according to claim **8**, wherein step (2) further comprises the step of:

(c) maintaining a current sequence number, an end of packet number, a highest sequence number and the expected sequence number in the data table; and

(d) setting the expected sequence number to the highest sequence number.

10. The method according to claim **1**, wherein step (2) comprises the step of:

(a) determining that a data packet is a valid data packet if the data packet is was not previously received and if the data packet has a sequence number that is less than a sequence number of a previously received data packet.

11. A system for eliminating unnecessary retransmissions of data packets from a first terminal, comprising:

a round trip timer that estimates a round trip time that it takes for a data packet to travel to a second terminal and for an acknowledgment of receipt to return from the second terminal, based upon delay in transmission plus throughput latency resulting from a slow link;

data tables that store information associated with the data packets including an expected sequence number; and

a discriminator that receives the data packets from the first terminal, that employs said round trip timer to determine whether the data packets are valid data packets or unnecessary retransmitted data packets, that forwards valid data packets to a second terminal and that discards unnecessary retransmitted data packets.

12. The system according to claim **11**, wherein said discriminator comprises:

means for determining that the data packet is a valid data packet when said round trip timer has expired.

13. The system according to claim **12**, wherein said discriminator comprises:

means for determining that the data packet is a valid data packet when said round trip timer has not expired and the data packet has a sequence number that is not less than the expected sequence number.

14. The system according to claim **13**, wherein said discriminator comprises:

means for determining that the data packet is a valid data packet when said round trip timer has not expired, the data packet has a sequence number that is less than the expected sequence number, and the data packet includes data that is not substantially similar to data in a previously received data packet.

15. The system according to claim **14**, wherein said discriminator further comprises:

means for maintaining a current sequence number, an end of packet number, a highest sequence number and the expected sequence number in said data table;

means for setting the end of packet number to a sum of a TCP sequence number of the data packet and an IP length field of the data packet;

means for setting the expected sequence number to the end of packet number when the sequence number is equal to the expected sequence number;

means for setting the highest sequence number to the end of packet number when the end of packet number is greater than the highest sequence number; and

means for setting the expected sequence number to the highest sequence number when the data packet has a sequence number that is less than the expected sequence number and the data packet includes data that is not substantially similar to data in a previously received data packet.

16. The system according to claim **11**, wherein said discriminator comprises:

means for determining that the data packet is an unnecessary retransmitted data packet when the data packet has a sequence number that is less than the expected sequence number and the data packet includes data that is substantially similar to data in a previously received data packet.

17. The method according to claim **16**, wherein said discriminator further comprises:

means for maintaining a current sequence number, an end of packet number, a highest sequence number and the expected sequence number in said data tables; and

means for setting the expected sequence number to the highest sequence number.

18. A computer program product comprising a computer useable medium having computer program logic stored therein, said computer program logic for enabling a computer to eliminate unnecessary retransmissions from a first terminal, wherein said computer program logic comprises:

means for enabling the computer to receive a data packet from the first terminal;

means for enabling the computer to determine whether the data packet is a valid data packet or an unnecessary retransmitted data packet, said determination considering a round trip time based upon delay in transmission plus throughput latency resulting from a slow link;

means for enabling the computer to forward valid data packets to a second terminal; and

means for enabling the computer to discard unnecessary retransmitted data packets.

19. The computer program product according to claim **18**, wherein said means for enabling the computer to determine whether the data packet is a valid data packet or an unnecessary retransmitted data packet comprises:

means for enabling the computer to estimate a round trip time that it takes for a data packet to travel to the second terminal and for an acknowledgment of receipt to return from the second terminal, based upon delay in transmission plus throughput latency resulting from a slow link; and

means for enabling the computer to determine whether the data packet is a valid data packet or an unnecessary retransmitted data packet based upon the estimated round trip time.

* * * * *